

Kokum

Complete Command & Example Reference

*From your first variable to concurrent,
networked and GUI applications.*



KOKUM 0.29.0

105 public built-ins · Language syntax · Native APIs
Small examples · Toolchain commands · Alphabetical lookup

Current baseline: Phase 4.17.7.1 + PUBLIC preservation fix
Documentation edition · 10 September 2026

About this reference

A practical desk reference for the stabilized Kokum programming language. This book changes no language features, source files, runtime formats or version numbers.

Reference scope	What is included
Language	75 focused language entries, including declarations, operators, collections, routines, classes, loops, exceptions and named tasks.
Built-ins	Every public function in the current catalogue: 105 public entries. Internal helper KOKUMFIND (catalogue slot 104) is deliberately excluded.
Resource APIs	112 native methods, 90 readable native properties, regular expressions, databases, images, graphs, WebSockets and synchronization.
GUI and tools	21 callable GUI method examples; 11 browser-API distinctions with Kokum alternatives; 78 toolchain command/option entries.
Practical integration	10 natural-pattern entries, four HTTP recipes, local service setups and a searchable, clickable alphabetical index.

How to find a command

Use the contents for a topic, or the A–Z index for a command or property. Every reference entry has a stable label such as BI-014 or API-011. Index entries link directly to that example and include its printed page number. Word's Navigation pane also follows the chapter and reference headings.

Read the context before copying

MAIN FRAGMENT means paste the code inside FUNCTION Main AS INTEGER, before RETURN 0. COMPLETE KOKUM SOURCE means save the listing as a complete .kokum file. GUI fragments belong inside a bound slot and require the named controls or components. Shell commands run in a terminal, not in Kokum code. SET- references give the shared prerequisites.

Evidence and boundaries

Examples were checked against the supplied baseline. Linux console, local HTTP/FlatDB, and native GUI-slot tests were executed. External TigerDB/PostgreSQL configuration, native Windows/macOS execution and browser layout behavior are not certified by those checks. The validation appendix records coverage and known distinctions precisely.

Kokum and the supplied mascot belong to the project's branding. Prepared for TigerDB Solutions Pvt. Ltd. Source documentation and catalogue paths appear in the provenance appendix. All test credentials and endpoint values are local demonstrations, not production secrets.

Contents

READ BY TOPIC

Start here: syntax and execution	4
Variables, values and collections	7
Decisions, loops and errors	12
Routines, modules and object-oriented code	15
Tasks, named threads and shared-state tools	21
General-purpose built-in functions	34
Mathematics and array statistics	41
Files, regular expressions and natural patterns	45
Forms, signals, slots and control APIs	53
Database commands and data components	63
Graph and image APIs	69
WebSocket APIs	75
HTTP GET, POST and microservices	82
Compiler, VM and IDE command line	86
Native property reference	99
Coverage, limitations and source provenance	105
Alphabetical command and example index	107

Quick routes

First program and the fragment wrapper · page 4
Module imports and aliases · page 18
Reusable reflection class · page 38
GUI slot and component setup · page 53
SQLite lab tables · page 65
Local WebSocket echo service · page 75
Local GET/POST service · page 82

01 / REFERENCE

Start here: syntax and execution

A common execution model makes the small examples easy to copy, run and compare.

SOURCE BASIS [S1] Core language contract, current parser and executable help.

A complete first program

```
FUNCTION Main AS INTEGER
  ? "Hello, Kokum"
  RETURN 0
ENDFUNC
```

Save this as hello.kokum in a writable lab directory. Put the current Kokum binaries on PATH, or prefix each command with its absolute path. The public compiler and IDE are Linux tools in this baseline.

```
kokumc check hello.kokum
kokumc run hello.kokum
kokumc run-ir hello.kokum
kokumc run-bytecode hello.kokum
kokumc compile hello.kokum -o hello.kbc
kokumvm hello.kbc
```

Each run prints Hello, Kokum. check validates source without running it. run-bytecode and kokumvm use the VM execution engine. A compiled .kbc is distinct from a project .kapp bundle.

The wrapper for a MAIN FRAGMENT

```
FUNCTION Main AS INTEGER
  && Paste one MAIN FRAGMENT here.
  RETURN 0
ENDFUNC
```

Do not paste executable statements above procedures or functions. Top-level declarations such as MODULE, EXPORT, PUBLIC and CLASS remain outside Main. A shared class or helper routine belongs outside Main too; SET- links identify those cases.

Core reading rules

Topic	Rule
Case	Identifiers and keywords are case-insensitive. String contents and map keys should use the spelling shown by their API.
Values	Normal arrays are one-based. TableView row/numeric-column indexes and ComboBox.SelectedIndex are zero-based. Item insertion/removal methods have their own documented indexes.
Lines	A semicolon at the end continues a Kokum statement. A shell uses backslash, not semicolon, for the shown multi-line commands.
Output	? prints values; multiple expressions are separated by commas. A Boolean printed directly appears as .T. or .F.; JSON uses true or false.
Safety	Use temporary files and the disposable local databases shown here. Do not run destructive sample SQL against a production database.

LANG-072 / ? (console output)

MAIN FRAGMENT

? expression [, expression ...]

Prints values followed by a newline. Use STR for explicit text formatting.

```
? "Hello, Kokum"
? 2 + 3, .T.
```

Result Prints Hello, Kokum, then 5 .T.

LANG-027 / Line continuation

MAIN FRAGMENT

```
statement ;
```

A semicolon at the end of a physical line continues the same statement on the next line. It is not a statement separator.

```
LOCAL total AS INTEGER
total = 10 + ;
      20 + ;
      30
? total
```

Result Prints 60.

LANG-028 / Comments

MAIN FRAGMENT

```
&& text, // text, -- text, leading # text, leading * text
```

Adds line comments that are ignored by the compiler.

```
&& comment
// comment
-- comment
* leading comment
? "comments ignored"
```

Result Prints comments ignored.

LANG-029 / Arithmetic operators

MAIN FRAGMENT

```
+ - * / %
```

Perform addition/concatenation, subtraction, multiplication, division, and remainder.

```
? 7 + 3, 7 - 3, 7 * 3, 8 / 2, 7 % 3
? "Ko" + "kum"
```

Result Prints 10, 4, 21, 4, 1, and Kokum.

LANG-030 / Comparison operators

MAIN FRAGMENT

```
= == != <> # < <= > >=
```

Compare values; = and == are equality, while !=, <>, and # are inequality forms.

```
? 5 = 5, 5 != 4, 3 < 4, 4 >= 4
```

Result Prints .T. for all four comparisons.

LANG-031 / Logical AND

MAIN FRAGMENT

```
left AND right / left .AND. right
```

Returns .T. only when both logical operands are .T.; evaluation short-circuits.

```
? .T. AND .F., .T. .AND. .T.
```

Result Prints .F. and .T.

LANG-032 / Logical OR

MAIN FRAGMENT

```
left OR right / left .OR. right
```

Returns .T. when either logical operand is .T.; evaluation short-circuits.

```
? .F. OR .T., .F. .OR. .F.
```

Result Prints .T. and .F.

LANG-033 / Logical NOT

MAIN FRAGMENT

NOT value / .NOT. value

Negates a logical value.

```
? NOT .F., .NOT. .T.
```

Result Prints .T. and .F.

LANG-034 / Grouping

MAIN FRAGMENT

(expression)

Controls expression precedence.

```
? (2 + 3) * 4
```

Result Prints 20.

02 / REFERENCE

Variables, values and collections

Declare intent clearly, initialize values explicitly and use the correct indexing convention.

SOURCE BASIS [S1–S3] Core language, value model, collections and decimal contract.

A typed declaration starts with NULL until assigned. PUBLIC is session/invocation state, not a process-wide shared array. PRIVATE is frame-owned in Kokum; do not assume historical XBase dynamic caller-chain lookup. Keep custom PUBLIC declarations outside the designer's marked generated blocks.

LANG-013 / LOCAL**COMPLETE KOKUM SOURCE**

LOCAL name [AS Type] [, ...]

Declares a routine-local variable.

```
FUNCTION Main AS INTEGER
  LOCAL total AS INTEGER
  total = 25
  ? total
  RETURN 0
ENDFUNC
```

Result Prints 25.

LANG-014 / PRIVATE**COMPLETE KOKUM SOURCE**

PRIVATE name [AS Type] [, ...]

Declares frame-owned private storage in the current core implementation.

```
FUNCTION Main AS INTEGER
  PRIVATE scratch AS NUMBER
  scratch = 12.5
  ? scratch
  RETURN 0
ENDFUNC
```

Result Prints 12.5.

LANG-015 / PUBLIC**COMPLETE KOKUM SOURCE**

PUBLIC name [AS Type] [, ...]

Declares program-wide storage for one interpreter/VM invocation.

```
PUBLIC appName AS STRING
FUNCTION Main AS INTEGER
  appName = "Kokum"
  ? appName
  RETURN 0
ENDFUNC
```

Result Prints Kokum.

LANG-016 / STATIC

COMPLETE KOKUM SOURCE

STATIC name [AS Type] [, ...]

Declares storage that persists between calls to the declaring routine or method.

```

FUNCTION NextNumber AS INTEGER
    STATIC n AS INTEGER
    IF TYPEOF(n) = "NULL"
        n = 0
    ENDIF
    n = n + 1
    RETURN n
ENDFUNC

FUNCTION Main AS INTEGER
    ? NextNumber(), NextNumber()
    RETURN 0
ENDFUNC

```

Result Prints 1 and 2.

LANG-017 / AS type annotation

MAIN FRAGMENT

name AS Type

Requests compile-time and runtime type checking for variables, parameters, results, and properties.

```

LOCAL price AS DECIMAL
price = 19.95M
? TYPEOF(price)

```

Result Prints DECIMAL.

LANG-018 / Assignment

MAIN FRAGMENT

target = expression

Stores a value in a variable, array element, map entry/member, or object property.

```

LOCAL values AS ARRAY
values = [10, 20]
values[2] = 25
? values[2]

```

Result Prints 25.

LANG-019 / Array literal

MAIN FRAGMENT

[expression, ...]

Creates a mutable ordered array.

```

LOCAL values AS ARRAY
values = [10, 20, 30]
? values[1], LEN(values)

```

Result Prints 10 and 3.

LANG-020 / Map literal

MAIN FRAGMENT

{key: value, ...}

Creates a mutable insertion-ordered map.

```

LOCAL person AS MAP
person = {"name": "Anil", "city": "Ratnagiri"}
? person["name"]

```

Result Prints Anil.

LANG-021 / Map dot member**MAIN FRAGMENT**

```
map.member
```

Uses a string-key shortcut for a map entry.

```
LOCAL m AS MAP
m = {"status": "ready"}
? m.status
m.status = "done"
? m.status
```

Result Prints ready and done.

LANG-022 / LOGICAL literals**MAIN FRAGMENT**

```
.T. / .F. / TRUE / FALSE
```

Creates Boolean values.

```
? .T., FALSE, TYPEOF(TRUE)
```

Result Prints .T., .F., and LOGICAL.

LANG-023 / NULL literals**MAIN FRAGMENT**

```
NULL / NIL / .NULL. / .NIL.
```

Creates the absence-of-value sentinel.

```
? TYPEOF(NULL), NULL = NIL
```

Result Prints NULL and .T.

LANG-024 / DECIMAL literal**MAIN FRAGMENT**

```
digits[.digits]M
```

Creates an exact base-10 decimal value.

```
LOCAL amount AS DECIMAL
amount = 199.95M
? amount + 0.05M
```

Result Prints 200. The arithmetic is exact DECIMAL; display omits insignificant trailing zeroes.

LANG-025 / DATE and DATETIME values**MAIN FRAGMENT**

```
DATE() / DATETIME()
```

Creates current local date and date-time values.

```
? TYPEOF(DATE()), TYPEOF(DATETIME())
```

Result Prints DATE and DATETIME.

LANG-026 / Code block literal**MAIN FRAGMENT**

```
{|parameter, ...| expression}
```

Creates a lexical expression closure.

```
LOCAL factor AS INTEGER
LOCAL twice AS CODEBLOCK
factor = 2
twice = {|x| x * factor}
? twice(5)
```

Result Prints 10.

LANG-035 / Index access

MAIN FRAGMENT

```
collection[indexOrKey]
```

Reads array elements by one-based index or map values by key.

```
? ["a", "b"][2]
? {"id": 7}["id"]
```

Result Prints b and 7.

LANG-036 / Member access

MAIN FRAGMENT

```
value.Member
```

Reads an object, native resource, or form-control member.

```
LOCAL counter AS ATOMICINTEGER
counter = ATOMICINTEGER(5)
? counter.Value
```

Result Prints 5.

LANG-074 / Typed declarations and aliases

MAIN FRAGMENT

```
LOCAL name AS type
```

Declare before use. MONEY aliases DECIMAL, BOOLEAN aliases LOGICAL, and PATTERN aliases REGEX. A typed but uninitialized variable holds NULL until assigned.

```
LOCAL amount AS MONEY
LOCAL active AS BOOLEAN
amount = 12.50M
active = .T.
? TYPEOF(amount), TYPEOF(active)
```

Result Prints DECIMAL LOGICAL.

LANG-075 / Nested arrays

MAIN FRAGMENT

```
matrix[row][column]
```

Arrays can contain arrays. Ordinary array indices start at 1 in both dimensions.

```
LOCAL matrix AS ARRAY
matrix = [[10, 20], [30, 40]]
matrix[2][1] = 35
? matrix[2][1]
```

Result Prints 35.

All 32 type-completion names

Name / aliases	Meaning	Example declaration
ANY	Dynamic value	LOCAL value AS ANY
ARRAY	One-based sequence	LOCAL items AS ARRAY
BOOLEAN / LOGICAL	Boolean aliases	LOCAL ready AS LOGICAL
CHART / GRAPH	Graph resource	PUBLIC graphSales AS GRAPH
CODEBLOCK	Expression closure	LOCAL fn AS CODEBLOCK
DATABASE / DB	Database component	PUBLIC dbLocal AS DATABASE
DATE	Date value	LOCAL today AS DATE
DATETIME	Date/time value	LOCAL stamp AS DATETIME

Name / aliases	Meaning	Example declaration
DECIMAL / MONEY	Exact decimal arithmetic	LOCAL price AS DECIMAL
FILE / FILEHANDLE	Owned file handle	LOCAL file AS FILE
IMAGE / PICTURE	Image resource	PUBLIC imgLogo AS IMAGE
INTEGER	Signed integer	LOCAL count AS INTEGER
JSON	JSON-compatible value	LOCAL payload AS JSON
MAP	Key/value collection	LOCAL row AS MAP
NUMBER	Floating-point value	LOCAL ratio AS NUMBER
OBJECT	Class instance value	LOCAL value AS OBJECT
STRING	UTF-8 text	LOCAL name AS STRING
TASK	Pending/completed task	LOCAL pending AS TASK
MUTEX	Mutual-exclusion handle	LOCAL gate AS MUTEX
ATOMICINTEGER	Shared atomic integer	LOCAL counter AS ATOMICINTEGER
CHANNEL	Bounded message channel	LOCAL queue AS CHANNEL
REGEX / PATTERN	Compiled regular expression	LOCAL rx AS REGEX
VOID	No result / signature context	PROCEDURE Notify() ... ENDPROC
WEBSOCKET / WSCLIENT	WebSocket resource	PUBLIC wsFeed AS WEBSOCKET

NOTE Declaring DATABASE, GRAPH, IMAGE or WEBSOCKET does not create a configured resource. The Designer/component host supplies those handles. For ARRAY, MAP and scalar aliases, assign a value before performing operations.

03 / REFERENCE

Decisions, loops and errors

Use structured branches and loops, then isolate failures with TRY/CATCH/FINALLY.

SOURCE BASIS [S1] Parser, execution engines and exception handling.

LOOP skips the remainder of the current loop iteration. EXIT and BREAK leave the loop. FOR EACH works from a snapshot of the collection at loop entry, so mutations should not be used to redefine that iteration sequence.

LANG-037 / IF / ELSEIF / ELSE / ENDIF**MAIN FRAGMENT**

```
IF condition ... [ELSEIF condition ...] [ELSE ...] ENDIF
```

Selects one branch using logical conditions.

```
LOCAL score AS INTEGER
score = 72
IF score >= 80
    ? "HIGH"
ELSEIF score >= 50
    ? "MEDIUM"
ELSE
    ? "LOW"
ENDIF
```

Result Prints MEDIUM.

LANG-038 / DO WHILE / ENDDO**MAIN FRAGMENT**

```
DO WHILE logicalExpression ... ENDDO
```

Repeats a block while its condition is .T..

```
LOCAL n AS INTEGER
n = 1
DO WHILE n <= 3
    ? n
    n = n + 1
ENDDO
```

Result Prints 1, 2, and 3.

LANG-039 / DO CASE / CASE / OTHERWISE / ENDCASE**MAIN FRAGMENT**

```
DO CASE CASE condition ... [OTHERWISE ...] ENDCASE
```

Selects the first matching CASE branch.

```
LOCAL n AS INTEGER
n = 2
DO CASE
CASE n = 1
    ? "one"
CASE n = 2
    ? "two"
OTHERWISE
    ? "other"
ENDCASE
```

Result Prints two.

LANG-040 / FOR / TO / STEP / NEXT**MAIN FRAGMENT**

```
FOR var = start TO end [STEP amount] ... NEXT [var]
```

Runs a counted loop.

```
LOCAL i AS INTEGER
FOR i = 1 TO 5 STEP 2
  ? i
NEXT i
```

Result Prints 1, 3, and 5.

LANG-041 / FOR EACH / IN / NEXT**MAIN FRAGMENT**

```
FOR EACH value IN collection ... NEXT
```

Iterates over collection values.

```
LOCAL value
FOR EACH value IN [10, 20, 30]
  ? value
NEXT
```

Result Prints 10, 20, and 30.

LANG-042 / FOR EACH key/index, value**MAIN FRAGMENT**

```
FOR EACH keyOrIndex, value IN collection ... NEXT
```

Iterates with a one-based array index or map key plus value.

```
LOCAL i, value
FOR EACH i, value IN ["a", "b"]
  ? i, value
NEXT
```

Result Prints 1 a, then 2 b.

LANG-043 / EXIT**MAIN FRAGMENT**

```
EXIT
```

Leaves the nearest loop.

```
LOCAL i AS INTEGER
FOR i = 1 TO 10
  IF i = 4
    EXIT
  ENDIF
  ? i
NEXT
```

Result Prints 1, 2, and 3.

LANG-044 / BREAK**MAIN FRAGMENT**

```
BREAK
```

Alias-style loop exit supported by the core parser.

```
LOCAL i AS INTEGER
FOR i = 1 TO 5
  IF i = 3
    BREAK
  ENDIF
  ? i
NEXT
```

Result Prints 1 and 2.

LANG-045 / LOOP

MAIN FRAGMENT

LOOP

Skips the remainder of the current loop iteration.

```
LOCAL i AS INTEGER
FOR i = 1 TO 4
    IF i = 2
        LOOP
    ENDIF
    ? i
NEXT
```

Result Prints 1, 3, and 4.

LANG-046 / TRY / CATCH / FINALLY / ENDTRY

MAIN FRAGMENT

TRY ... [CATCH errorVar ...] [FINALLY ...] ENDTRY

Handles catchable runtime errors and guarantees cleanup.

```
LOCAL zero AS INTEGER
zero = 0
TRY
    ? 10 / zero
CATCH error
    ? "caught", TYPEOF(error)
FINALLY
    ? "cleanup"
ENDTRY
```

Result Prints caught ERROR and cleanup.

04 / REFERENCE

Routines, modules and object-oriented code

Organize reusable code with functions, procedures, explicit imports and exported symbols.

SOURCE BASIS [S1, S4] Module, object and closure contracts.

LANG-006 / FUNCTION / ENDFUNC**COMPLETE KOKUM SOURCE**

```
FUNCTION Name([parameters]) [AS Type] ... ENDFUNC
```

Declares a value-returning routine.

```

FUNCTION Add(a AS INTEGER, b AS INTEGER) AS INTEGER
    RETURN a + b
ENDFUNC

FUNCTION Main AS INTEGER
    ? Add(2, 3)
    RETURN 0
ENDFUNC

```

Result Prints 5.

LANG-007 / PROCEDURE / ENDPROC**COMPLETE KOKUM SOURCE**

```
PROCEDURE Name([parameters]) ... ENDPROC
```

Declares a routine used primarily for side effects; successful completion returns NULL.

```

PROCEDURE SayHello(name AS STRING)
    ? "Hello " + name
ENDPROC

FUNCTION Main AS INTEGER
    SayHello("Anil")
    RETURN 0
ENDFUNC

```

Result Prints Hello Anil.

LANG-010 / PARAMETERS**COMPLETE KOKUM SOURCE**

```
PARAMETERS name1 [, name2 ...]
```

Declares untyped body parameters at the start of a routine.

```

FUNCTION Multiply
    PARAMETERS a, b
    RETURN a * b
ENDFUNC

FUNCTION Main AS INTEGER
    ? Multiply(3, 4)
    RETURN 0
ENDFUNC

```

Result Prints 12.

LANG-011 / LPARAMETERS

COMPLETE KOKUM SOURCE

LPARAMETERS name [AS Type] [, ...]

Declares typed or untyped local body parameters at the start of a routine.

```

FUNCTION Describe
    LPARAMETERS name AS STRING, count AS INTEGER
    RETURN name + ": " + STR(count)
ENDFUNC

FUNCTION Main AS INTEGER
    ? Describe("items", 3)
    RETURN 0
ENDFUNC

```

Result Prints items: 3.

LANG-012 / RETURN

COMPLETE KOKUM SOURCE

RETURN [expression]

Ends the current routine; functions may return a value and procedures may return without one.

```

FUNCTION Answer AS INTEGER
    RETURN 42
ENDFUNC

FUNCTION Main AS INTEGER
    ? Answer()
    RETURN 0
ENDFUNC

```

Result Prints 42.

LANG-047 / CLASS / ENDCLASS

COMPLETE KOKUM SOURCE

CLASS Name [AS BaseClass] ... ENDCLASS

Declares a class, optionally with single inheritance.

```

CLASS Person
    PROPERTY Name AS STRING = "Anil"
ENDCLASS

FUNCTION Main AS INTEGER
    LOCAL p AS Person
    p = NEW Person()
    ? p.Name
    RETURN 0
ENDFUNC

```

Result Prints Anil.

LANG-048 / PROPERTY

COMPLETE KOKUM SOURCE

PROPERTY Name [AS Type] [= default]

Declares per-object state.

```

CLASS Counter
  PROPERTY Value AS INTEGER = 0
ENDCLASS

FUNCTION Main AS INTEGER
  LOCAL c AS Counter
  c = NEW Counter()
  c.Value = 3
  ? c.Value
  RETURN 0
ENDFUNC

```

Result Prints 3.

LANG-049 / METHOD / ENDMETHOD

COMPLETE KOKUM SOURCE

METHOD Name([parameters]) [AS Type] ... ENDMETHOD

Declares class behavior.

```

CLASS Person
  PROPERTY Name AS STRING = "Anil"
  METHOD Greet AS STRING
    RETURN "Hello " + THIS.Name
  ENDMETHOD
ENDCLASS

FUNCTION Main AS INTEGER
  ? NEW Person().Greet()
  RETURN 0
ENDFUNC

```

Result Prints Hello Anil.

LANG-050 / THIS

COMPLETE KOKUM SOURCE

THIS.Member

Refers to the current object inside a method.

```

CLASS Box
  PROPERTY Width AS INTEGER = 4
  METHOD Area AS INTEGER
    RETURN THIS.Width * THIS.Width
  ENDMETHOD
ENDCLASS

FUNCTION Main AS INTEGER
  ? NEW Box().Area()
  RETURN 0
ENDFUNC

```

Result Prints 16.

LANG-051 / NEW

COMPLETE KOKUM SOURCE

```
NEW ClassName([arguments])
```

Constructs an object and invokes an effective Init method when present.

```

CLASS Person
  PROPERTY Name AS STRING
  METHOD Init(name AS STRING)
    THIS.Name = name
  ENDMETHOD
ENDCLASS

FUNCTION Main AS INTEGER
  LOCAL p AS Person
  p = NEW Person("Anil")
  ? p.Name
  RETURN 0
ENDFUNC

```

Result Prints Anil.

LANG-052 / Single inheritance with AS

COMPLETE KOKUM SOURCE

```
CLASS Child AS Parent
```

Inherits properties and methods from one base class.

```

CLASS Animal
  METHOD Speak AS STRING
    RETURN "sound"
  ENDMETHOD
ENDCLASS

CLASS Dog AS Animal
ENDCLASS

FUNCTION Main AS INTEGER
  ? NEW Dog().Speak()
  RETURN 0
ENDFUNC

```

Result Prints sound.

NOTE Inheritance is single-base. There is no multiple-inheritance syntax or implicit Java/C# compatibility.

A complete two-module lab

Create the following three files in a new directory. The alias AddTwo deliberately avoids SUM, which is already a built-in. The optional Trace import is not invoked; calling an unavailable optional import raises an error.

math.kokum

```

MODULE sample.math VERSION "1.0.0"
EXPORT FUNCTION InternalTotal AS Add

FUNCTION InternalTotal(a AS INTEGER, b AS INTEGER) AS INTEGER
  RETURN a + b
ENDFUNC

```

app.kokum

```

MODULE sample.app VERSION "1.0.0"
IMPORT FUNCTION Add AS AddTwo FROM sample.math VERSION "^1.0.0"
IMPORT OPTIONAL FUNCTION Trace FROM tools.trace
EXPORT FUNCTION Main

FUNCTION Main AS INTEGER
  ? AddTwo(2, 3)
  RETURN 0
ENDFUNC

```

kokum.toml

```
[project]
name = "ReferenceModules"
version = "1.0.0"
kind = "console"
entry_module = "sample.app"
entry = "Main"

[build]
output_dir = "build"

[bundle]
output = "demo.kapp"
target_os = "portable"
target_arch = "any"

[[module]]
name = "sample.math"
version = "1.0.0"
source = "math.kokum"

[[module]]
name = "sample.app"
version = "1.0.0"
source = "app.kokum"
```

Run in the terminal

```
kokumc project check kokum.toml
kokumc build kokum.toml
kokumc bundle kokum.toml
kokumvm demo.kapp
```

The linked application prints 5. Module names and versions in the manifest match the declarations in the source files.

LANG-001 / MODULE**MODULE PROJECT (SEE SETUP) · SET-MODULE**

```
MODULE module.name [VERSION "x.y.z"]
```

Declares the logical module represented by the source file.

```
MODULE sample.math VERSION "1.0.0"
```

Result The file is compiled as module sample.math version 1.0.0.

NOTE First line of math.kokum in SET-MODULE.

LANG-002 / VERSION**MODULE PROJECT (SEE SETUP) · SET-MODULE**

```
MODULE name VERSION "major.minor.patch"
```

Assigns a semantic version to a module.

```
MODULE sample.math VERSION "1.0.0"
```

Result The module identity is sample.math@1.0.0; match the project manifest.

LANG-003 / IMPORT**MODULE PROJECT (SEE SETUP) · SET-MODULE**

```
IMPORT [OPTIONAL] kind Name [AS Alias] FROM module [VERSION "constraint"]
```

Imports an exported symbol, optionally under a local alias. Required imports must resolve when linking.

```
IMPORT FUNCTION Add AS AddTwo FROM sample.math VERSION "^1.0.0"
```

Result Calls to AddTwo() resolve to the exported Add function.

NOTE Do not use SUM as the alias: it already names a built-in function. The complete two-module lab uses AddTwo.

LANG-004 / OPTIONAL

MODULE PROJECT (SEE SETUP) · SET-MODULE

```
IMPORT OPTIONAL ...
```

Allows linking to continue when an import is unavailable; calling it later raises a catchable error.

```
IMPORT OPTIONAL FUNCTION Trace FROM tools.trace
```

Result The module links without tools.trace; Trace() is checked at runtime.

LANG-005 / EXPORT

MODULE PROJECT (SEE SETUP) · SET-MODULE

```
EXPORT kind LocalName [AS PublicName]
```

Publishes a local declaration from a module.

```
EXPORT FUNCTION InternalTotal AS Add
```

Result The local InternalTotal function is exported as Add.

NOTE IMPORT/EXPORT metadata also recognizes CLASS and GLOBAL, and compatibility import kinds include BUILTIN/NATIVE. Cross-module mutable-global execution and arbitrary dynamic native-library loading are not promised by this baseline; prefer the supported function/procedure module pattern.

05 / REFERENCE

Tasks, named threads and shared-state tools

Keep the owning session responsive while workers do independent work.

SOURCE BASIS [S5] ASYNC/TASK, named-thread and synchronization implementations.

Real overlap is provided by KokumVM and run-bytecode. Ordinary AST/IR ASYNC reference execution is inline; source using named-thread lowering can route through the VM. AWAIT, Result and Wait block the waiting session. A wait timeout is not cancellation and does not justify retrying a POST.

Shared helper declarations

```
PROCEDURE Work()
  SLEEP(1)
ENDPROC

PROCEDURE SlowWork()
  SLEEP(50)
ENDPROC

PROCEDURE FirstStep()
  SLEEP(1)
ENDPROC

PROCEDURE SecondStep()
  ? "second"
ENDPROC
```

Place these helpers above Main for entries that link to SET-THREAD. They are deliberately small so the examples focus on scheduling and ownership.

LANG-008 / ASYNC FUNCTION

COMPLETE KOKUM SOURCE

```
ASYNC FUNCTION Name(...) [AS ResultType] ... ENDFUNC
```

Calling an ASYNC function produces a TASK. KokumVM executes on the bounded worker pool; reference AST/IR execution is inline.

```
ASYNC FUNCTION Double(n AS INTEGER) AS INTEGER
  RETURN n * 2
ENDFUNC

FUNCTION Main AS INTEGER
  LOCAL task AS TASK
  task = Double(21)
  ? AWAIT task
  RETURN 0
ENDFUNC
```

Result Prints 42.

LANG-009 / ASYNC PROCEDURE

COMPLETE KOKUM SOURCE

```
ASYNC PROCEDURE Name(...) ... ENDPROC
```

An ASYNC procedure returns a TASK whose successful result is NULL. Await it when completion matters.

```
ASYNC PROCEDURE PauseBriefly()
  SLEEP(10)
ENDPROC

FUNCTION Main AS INTEGER
  AWAIT PauseBriefly()
  ? "done"
  RETURN 0
ENDFUNC
```

Result Prints done after the task completes.

LANG-073 / ASYNC METHOD

COMPLETE KOKUM SOURCE

```
ASYNC METHOD Name(...) [AS ResultType] ... ENDMETHOD
```

An asynchronous instance method produces a TASK. Receivers are snapshotted for the worker.

```
CLASS Calculator
  ASYNC METHOD Twice(n AS INTEGER) AS INTEGER
    RETURN n * 2
  ENDMETHOD
ENDCLASS

FUNCTION Main AS INTEGER
  LOCAL c AS Calculator
  c = NEW Calculator()
  ? AWAIT c.Twice(9)
  RETURN 0
ENDFUNC
```

Result Prints 18.

LANG-060 / AWAIT

COMPLETE KOKUM SOURCE

```
AWAIT taskExpression
```

Waits for a TASK and returns its value or rethrows its error.

```
ASYNC FUNCTION Square(n AS INTEGER) AS INTEGER
  RETURN n * n
ENDFUNC

FUNCTION Main AS INTEGER
  ? AWAIT Square(6)
  RETURN 0
ENDFUNC
```

Result Prints 36.

LANG-061 / RUN ... AS

COMPLETE KOKUM SOURCE

```
RUN Routine(arguments) AS threadName
```

Starts a named logical task.

```
PROCEDURE Work()
  ? "work"
ENDPROC

FUNCTION Main AS INTEGER
  RUN Work() AS worker
  WAIT FOR worker
  RETURN 0
ENDFUNC
```

Result Starts Work and waits for it.

LANG-062 / NOWAIT

MAIN FRAGMENT + SHARED DECLARATION · SET-THREAD

```
RUN ... AS name NOWAIT
```

Excludes a named task from automatic scope joining.

```
RUN Work() AS worker NOWAIT
? "owner continues"
WAIT FOR worker
```

Result The owner continues immediately, then explicitly joins worker.

NOTE Include the Work procedure printed in the named-task setup.

LANG-063 / Result publication (:name)**COMPLETE KOKUM SOURCE**

```
RUN Function(...) AS thread NOWAIT :resultVar
```

Publishes the task result into a variable when the task is awaited.

```
FUNCTION Add(a, b)
  RETURN a + b
ENDFUNC

FUNCTION Main AS INTEGER
  LOCAL answer
  RUN Add(10, 20) AS calc NOWAIT :answer
  WAIT FOR calc
  ? answer
  RETURN 0
ENDFUNC
```

Result Prints 30.

LANG-064 / Dependency WAIT FOR on RUN**MAIN FRAGMENT + SHARED DECLARATION · SET-THREAD**

```
RUN Second() AS second WAIT FOR first
```

Starts a task only after the named dependency completes.

```
RUN FirstStep() AS first
RUN SecondStep() AS second WAIT FOR first NOWAIT
WAIT FOR second
```

Result SecondStep begins after FirstStep completes.

NOTE FirstStep and SecondStep are the ordinary procedures in the named-task setup.

LANG-065 / WAIT FOR task**MAIN FRAGMENT + SHARED DECLARATION · SET-THREAD**

```
WAIT FOR threadName
```

Waits for one named task.

```
RUN Work() AS worker NOWAIT
WAIT FOR worker
? "joined"
```

Result Prints joined after worker completes.

NOTE Include the Work procedure printed in the named-task setup.

LANG-066 / WAIT FOR task list**MAIN FRAGMENT + SHARED DECLARATION · SET-THREAD**

```
WAIT FOR name1, name2 [, ...]
```

Waits for all listed named tasks.

```
RUN Work() AS one NOWAIT
RUN Work() AS two NOWAIT
WAIT FOR one, two
```

Result Both tasks are complete when execution continues.

NOTE Include the Work procedure printed in the named-task setup.

LANG-067 / WAIT FOR ALL

MAIN FRAGMENT + SHARED DECLARATION · SET-THREAD

WAIT FOR ALL

Waits for all outstanding named tasks in the current scope.

```

RUN Work() AS one NOWAIT
RUN Work() AS two NOWAIT
WAIT FOR ALL

```

Result All outstanding named tasks are complete.

NOTE Include the Work procedure printed in the named-task setup.

LANG-068 / WAIT FOR ... TIMEOUT

MAIN FRAGMENT + SHARED DECLARATION · SET-THREAD

WAIT FOR name [,...] TIMEOUT milliseconds

Performs a bounded wait; timeout is catchable.

```

RUN SlowWork() AS worker NOWAIT
TRY
    WAIT FOR worker TIMEOUT 5
CATCH error
    ? "timeout"
ENDTRY
WAIT FOR worker

```

Result Prints timeout if work exceeds 5 ms, then completes the final join.

NOTE SlowWork sleeps for 50 ms; the timeout does not cancel it.

LANG-069 / LOCK ... ENDLOCK

MAIN FRAGMENT

LOCK mutexVariable [TIMEOUT ms] ... ENDLOCK

Locks a simple mutex variable for a lexical block and releases it during cleanup, including errors and returns.

```

LOCAL gate AS MUTEX
gate = MUTEX()
LOCK gate TIMEOUT 1000
    ? "protected"
ENDLOCK

```

Result Prints protected and releases gate.

Task and synchronization built-ins

BI-038 / SLEEP

MAIN FRAGMENT

SLEEP(milliseconds)

Blocks the current execution for the requested milliseconds.

```

? "before"
SLEEP(50)
? "after"

```

Result Prints before, pauses briefly, then prints after.

BI-072 / THREADSTATUS

MAIN FRAGMENT + SHARED DECLARATION · SET-THREAD

THREADSTATUS(task)

Returns the named task status string.

```

RUN SlowWork() AS worker NOWAIT
? THREADSTATUS(worker)
WAIT FOR worker

```

Result Prints a state such as QUEUED, RUNNING or COMPLETED; the first state depends on scheduling.

NOTE Use SlowWork from SET-THREAD. THREADSTATUS uses uppercase state names; task.Status uses lowercase names.

BI-073 / THREADDONE

MAIN FRAGMENT + SHARED DECLARATION · SET-THREAD

THREADDONE(task)

Tests whether a named task has finished.

```

RUN SlowWork() AS worker NOWAIT
? THREADDONE(worker)
WAIT FOR worker
? THREADDONE(worker)

```

Result The final result is .T.

NOTE Use SlowWork from SET-THREAD; the final check follows WAIT FOR.

BI-074 / THREADERROR

MAIN FRAGMENT + SHARED DECLARATION · SET-THREAD

THREADERROR(task)

Returns a named task error message, or an empty string when none exists.

```

RUN Work() AS worker NOWAIT
WAIT FOR worker
? THREADERROR(worker)

```

Result Prints an empty string after successful work.

NOTE Use Work from SET-THREAD. An empty message means no task fault.

BI-075 / THREADID

MAIN FRAGMENT + SHARED DECLARATION · SET-THREAD

THREADID(task)

Returns Kokum's worker-thread identifier for the named task.

```

RUN Work() AS worker NOWAIT
WAIT FOR worker
? THREADID(worker)

```

Result Prints a positive worker identifier after the task has run.

NOTE Use Work from SET-THREAD. This is a Kokum platform thread identifier, not a unique task ID; tasks can reuse a worker.

BI-076 / THREADPOOLSIZE

MAIN FRAGMENT

THREADPOOLSIZE()

Returns the configured bounded worker-pool size.

```

? THREADPOOLSIZE()

```

Result Prints a positive integer.

NOTE All tasks in the process share the configured bounded worker pool.

BI-077 / MUTEX**MAIN FRAGMENT**

```
MUTEX()
```

Creates a MUTEX native resource.

```
LOCAL gate AS MUTEX
gate = MUTEX()
? TYPEOF(gate)
```

Result Prints MUTEX.

BI-078 / LOCK**MAIN FRAGMENT**

```
LOCK(mutex [, timeoutMs])
```

Function form: acquires a MUTEX, optionally with timeout, and returns .T..

```
LOCAL gate AS MUTEX
gate = MUTEX()
? LOCK(gate, 1000)
UNLOCK(gate)
```

Result Prints .T. when the mutex is acquired.

BI-079 / UNLOCK**MAIN FRAGMENT**

```
UNLOCK(mutex)
```

Function form: releases a MUTEX owned by the current execution.

```
LOCAL gate AS MUTEX
gate = MUTEX()
LOCK(gate)
? UNLOCK(gate)
```

Result Prints .T.

BI-080 / MUTEXSTATUS**MAIN FRAGMENT**

```
MUTEXSTATUS(mutex)
```

Returns mutex ownership and contention diagnostics.

```
LOCAL gate AS MUTEX
gate = MUTEX()
? JSONENCODE(MUTEXSTATUS(gate))
```

Result Prints a status map.

BI-081 / ATOMICINTEGER**MAIN FRAGMENT**

```
ATOMICINTEGER([initial])
```

Creates an atomic signed integer, optionally with an initial value.

```
LOCAL counter AS ATOMICINTEGER
counter = ATOMICINTEGER(10)
? counter.Value
```

Result Prints 10.

BI-082 / ATOMICGET**MAIN FRAGMENT**

ATOMICGET(counter)

Reads an atomic integer.

```
LOCAL counter AS ATOMICINTEGER
counter = ATOMICINTEGER(10)
? ATOMICGET(counter)
```

Result Prints 10.

BI-083 / ATOMICSET**MAIN FRAGMENT**

ATOMICSET(counter, value)

Sets an atomic value and returns the previous value.

```
LOCAL counter AS ATOMICINTEGER
counter = ATOMICINTEGER(10)
? ATOMICSET(counter, 20), ATOMICGET(counter)
```

Result Prints 10 and 20.

BI-084 / ATOMICADD**MAIN FRAGMENT**

ATOMICADD(counter, delta)

Adds a delta atomically and returns the new value.

```
LOCAL counter AS ATOMICINTEGER
counter = ATOMICINTEGER(10)
? ATOMICADD(counter, 5)
```

Result Prints 15.

BI-085 / ATOMICINC**MAIN FRAGMENT**

ATOMICINC(counter)

Increments atomically and returns the new value.

```
LOCAL counter AS ATOMICINTEGER
counter = ATOMICINTEGER(0)
? ATOMICINC(counter)
```

Result Prints 1.

BI-086 / ATOMICDEC**MAIN FRAGMENT**

ATOMICDEC(counter)

Decrements atomically and returns the new value.

```
LOCAL counter AS ATOMICINTEGER
counter = ATOMICINTEGER(2)
? ATOMICDEC(counter)
```

Result Prints 1.

BI-087 / ATOMICCOMPAREEXCHANGE**MAIN FRAGMENT**

ATOMICCOMPAREEXCHANGE(counter, expected, desired)

Replaces the value only when it equals expected.

```
LOCAL counter AS ATOMICINTEGER
counter = ATOMICINTEGER(5)
? ATOMICCOMPAREEXCHANGE(counter, 5, 9), counter.Value
```

Result Prints .T. and 9.

BI-088 / CHANNEL**MAIN FRAGMENT**

CHANNEL(capacity)

Creates a bounded transferable message channel.

```
LOCAL queue AS CHANNEL
queue = CHANNEL(2)
? queue.Capacity
```

Result Prints 2.

BI-089 / CHANNELSEND**MAIN FRAGMENT**

CHANNELSEND(channel, value [, timeoutMs])

Sends one copied value, optionally with a timeout.

```
LOCAL queue AS CHANNEL
queue = CHANNEL(1)
? CHANNELSEND(queue, "hello", 1000)
```

Result Prints .T.

BI-090 / CHANNELRECEIVE**MAIN FRAGMENT**

CHANNELRECEIVE(channel [, timeoutMs])

Receives one value, optionally with a timeout; closed and drained returns NULL.

```
LOCAL queue AS CHANNEL
queue = CHANNEL(1)
CHANNELSEND(queue, 42)
? CHANNELRECEIVE(queue)
```

Result Prints 42.

BI-091 / CHANNELCLOSE**MAIN FRAGMENT**

CHANNELCLOSE(channel)

Closes a channel and wakes blocked senders/receivers.

```
LOCAL queue AS CHANNEL
queue = CHANNEL(1)
? CHANNELCLOSE(queue)
```

Result Prints .T.

BI-092 / CHANNELCOUNT**MAIN FRAGMENT**

CHANNELCOUNT(channel)

Returns the current queued-message count.

```
LOCAL queue AS CHANNEL
queue = CHANNEL(2)
CHANNELSEND(queue, 1)
? CHANNELCOUNT(queue)
```

Result Prints 1.

BI-093 / CHANNELCAPACITY**MAIN FRAGMENT**

```
CHANNELCAPACITY(channel)
```

Returns the configured channel capacity.

```
LOCAL queue AS CHANNEL
queue = CHANNEL(3)
? CHANNELCAPACITY(queue)
```

Result Prints 3.

BI-094 / CHANNELCLOSED**MAIN FRAGMENT**

```
CHANNELCLOSED(channel)
```

Tests whether a channel is closed.

```
LOCAL queue AS CHANNEL
queue = CHANNEL(1)
CHANNELCLOSE(queue)
? CHANNELCLOSED(queue)
```

Result Prints .T.

BI-095 / CHANNELSTATUS**MAIN FRAGMENT**

```
CHANNELSTATUS(channel)
```

Returns capacity, count, waiters, and counters.

```
LOCAL queue AS CHANNEL
queue = CHANNEL(2)
? JSONENCODE(CHANNELSTATUS(queue))
```

Result Prints a status map.

BI-096 / SYNCSTATS**MAIN FRAGMENT**

```
SYNCSTATS([resource])
```

Returns synchronization diagnostics globally or for one sync resource.

```
LOCAL counter AS ATOMICINTEGER
counter = ATOMICINTEGER(0)
? TYPEOF(SYNCSTATS(counter))
```

Result Prints MAP.

TASK method setup

```
ASYNC FUNCTION Double(n AS INTEGER) AS INTEGER
    RETURN n * 2
ENDFUNC
```

Place Double above Main. The method examples below create their own task. Results are cached in the owner session: reading the same task again does not repeat the computation or HTTP request.

API-082 / TASK.Await**MAIN FRAGMENT + SHARED DECLARATION · SET-TASK**

```
Await([timeoutMs AS INTEGER])
```

Wait for completion and return or rethrow the result.

```
LOCAL task AS TASK
task = Double(21)
? task.Await(3000)
```

Result Waits up to 3 seconds and prints the task result.

API-083 / TASK.Wait**MAIN FRAGMENT + SHARED DECLARATION · SET-TASK**

```
Wait([timeoutMs AS INTEGER])
```

Alias for Await.

```
LOCAL task AS TASK
task = Double(21)
? task.Wait(3000)
```

Result Alias of Await; prints the result.

API-084 / TASK.Result**MAIN FRAGMENT + SHARED DECLARATION · SET-TASK**

```
Result([timeoutMs AS INTEGER])
```

Alias for Await.

```
LOCAL task AS TASK
task = Double(21)
? task.Result(3000)
```

Result Alias of Await; prints the result.

API-085 / MUTEX.Lock**MAIN FRAGMENT**

```
Lock([timeoutMs AS INTEGER])
```

Acquire the mutex, optionally with a timeout.

```
LOCAL gate AS MUTEX
gate = MUTEX()
? gate.Lock(1000)
gate.Unlock()
```

Result Prints .T. when gate is acquired.

API-086 / MUTEX.TryLock**MAIN FRAGMENT**

```
TryLock([timeoutMs AS INTEGER])
```

Attempt to acquire the mutex without an unbounded wait.

```
LOCAL gate AS MUTEX
gate = MUTEX()
? gate.TryLock(0)
gate.Unlock()
```

Result Immediately attempts acquisition and prints .T./.F.

API-087 / MUTEX.Unlock**MAIN FRAGMENT**

```
Unlock()
```

Release a mutex owned by the current Kokum execution.

```
LOCAL gate AS MUTEX
gate = MUTEX()
gate.Lock()
? gate.Unlock()
```

Result Prints .T. when the current execution releases gate.

API-088 / MUTEX.Status**MAIN FRAGMENT**

```
Status()
```

Return mutex ownership and contention diagnostics.

```
LOCAL gate AS MUTEX
gate = MUTEX()
? JSONENCODE(gate.Status())
```

Result Prints ownership and contention diagnostics.

API-089 / ATOMICINTEGER.Get**MAIN FRAGMENT**

```
Get()
```

Read the current value.

```
LOCAL counter AS ATOMICINTEGER
counter = ATOMICINTEGER(5)
? counter.Get()
```

Result Prints the current atomic value.

API-090 / ATOMICINTEGER.Set**MAIN FRAGMENT**

```
Set(value AS INTEGER)
```

Set a value and return the previous value.

```
LOCAL counter AS ATOMICINTEGER
counter = ATOMICINTEGER(5)
? counter.Set(10)
```

Result Prints the previous value and stores 10.

API-091 / ATOMICINTEGER.Exchange**MAIN FRAGMENT**

```
Exchange(value AS INTEGER)
```

Atomically replace the value and return the previous value.

```
LOCAL counter AS ATOMICINTEGER
counter = ATOMICINTEGER(5)
? counter.Exchange(20)
```

Result Prints the previous value and stores 20.

API-092 / ATOMICINTEGER.Add**MAIN FRAGMENT**

```
Add(delta AS INTEGER)
```

Add a delta and return the new value.

```
LOCAL counter AS ATOMICINTEGER
counter = ATOMICINTEGER(5)
? counter.Add(5)
```

Result Adds 5 and prints the new value.

API-093 / ATOMICINTEGER.Increment**MAIN FRAGMENT**

```
Increment()
```

Increment and return the new value.

```
LOCAL counter AS ATOMICINTEGER
counter = ATOMICINTEGER(5)
? counter.Increment()
```

Result Increments and prints the new value.

API-094 / ATOMICINTEGER.Decrement**MAIN FRAGMENT**

```
Decrement()
```

Decrement and return the new value.

```
LOCAL counter AS ATOMICINTEGER
counter = ATOMICINTEGER(5)
? counter.Decrement()
```

Result Decrements and prints the new value.

API-095 / ATOMICINTEGER.CompareExchange**MAIN FRAGMENT**

```
CompareExchange(expected AS INTEGER, desired AS INTEGER)
```

Replace the value only when it equals expected.

```
LOCAL counter AS ATOMICINTEGER
counter = ATOMICINTEGER(5)
? counter.CompareExchange(5, 9)
```

Result Prints .T. and stores 9 only when current value is 5.

API-096 / ATOMICINTEGER.Status**MAIN FRAGMENT**

```
Status()
```

Return value and operation diagnostics.

```
LOCAL counter AS ATOMICINTEGER
counter = ATOMICINTEGER(5)
? JSONENCODE(counter.Status())
```

Result Prints value and operation diagnostics.

API-097 / CHANNEL.Send**MAIN FRAGMENT**

```
Send(value AS ANY [, timeoutMs AS INTEGER])
```

Send one copied message, waiting for capacity when needed.

```
LOCAL queue AS CHANNEL
queue = CHANNEL(2)
? queue.Send("hello", 1000)
```

Result Copies and sends hello; prints .T.

API-098 / CHANNEL.TrySend**MAIN FRAGMENT**

```
TrySend(value AS ANY)
```

Attempt an immediate send.

```
LOCAL queue AS CHANNEL
queue = CHANNEL(2)
? queue.TrySend("hello")
```

Result Attempts an immediate send; prints .T./F.

API-099 / CHANNEL.Receive

MAIN FRAGMENT

```
Receive([timeoutMs AS INTEGER])
```

Receive one message, optionally with a timeout.

```
LOCAL queue AS CHANNEL
queue = CHANNEL(2)
queue.Send(42)
? queue.Receive(1000)
```

Result Prints one received value, NULL after closed/drained, or times out.

API-100 / CHANNEL.TryReceive

MAIN FRAGMENT

```
TryReceive()
```

Return {Received, Value} without waiting.

```
LOCAL queue AS CHANNEL
queue = CHANNEL(2)
queue.Send("hello")
? JSONENCODE(queue.TryReceive())
```

Result Prints a map with Received and Value.

API-101 / CHANNEL.Close

MAIN FRAGMENT

```
Close()
```

Close the channel and wake blocked senders and receivers.

```
LOCAL queue AS CHANNEL
queue = CHANNEL(2)
? queue.Close()
```

Result Closes the channel and prints .T.

API-102 / CHANNEL.Status

MAIN FRAGMENT

```
Status()
```

Return capacity, queue depth, waiters, and counters.

```
LOCAL queue AS CHANNEL
queue = CHANNEL(2)
? JSONENCODE(queue.Status())
```

Result Prints capacity, depth, waiters, and counters.

Operational settings and ownership

```
export KOKUM_TASK_WORKERS=4
export KOKUM_TASK_QUEUE_CAPACITY=256
export KOKUM_TASK_SUBMIT_TIMEOUT_MS=5000
export KOKUM_URL_MAX_CONCURRENT=16
```

Set these before launching the VM. HTTP concurrency is process-wide; task workers and queued work are separate bounds. Ordinary ARRAY/MAP task arguments are snapshots, not shared mutable containers. Synchronization resources are the explicit shared-state tools. Keep GUI controls, files and session-owned resources on their owner thread. Retain and drain tasks whose outcomes matter.

06 / REFERENCE

General-purpose built-in functions

Conversion, text, collections, JSON, reflection and managed-runtime diagnostics.

SOURCE BASIS [S2, S4, S6] Public built-in catalogue and corresponding implementations.

Signatures use square brackets to denote optional arguments; do not type those brackets as part of the call. Native Kokum arrays and strings are not interchangeable with the response maps returned by other APIs. Function aliases listed separately are genuine catalogue entries.

BI-001 / STR

MAIN FRAGMENT

```
STR(value [, width [, decimals]])
```

Converts a value to text; optional width/decimals format numeric output.

```
? STR(125)
```

Result Prints 125.

BI-002 / LEN

MAIN FRAGMENT

```
LEN(value)
```

Returns the length of a string, array, or map.

```
? LEN("Kokum"), LEN([10, 20, 30])
```

Result Prints 5 and 3.

BI-003 / UPPER

MAIN FRAGMENT

```
UPPER(text)
```

Converts text to uppercase.

```
? UPPER("Kokum")
```

Result Prints KOKUM.

BI-004 / LOWER

MAIN FRAGMENT

```
LOWER(text)
```

Converts text to lowercase.

```
? LOWER("KOKUM")
```

Result Prints kokum.

BI-005 / ABS

MAIN FRAGMENT

```
ABS(number)
```

Returns the absolute numeric value.

```
? ABS(-12.5)
```

Result Prints 12.5.

BI-006 / INT**MAIN FRAGMENT**

INT(number)

Converts a numeric value to an integer by truncation toward zero.

```
? INT(7.9)
```

Result Prints 7.

BI-007 / VAL**MAIN FRAGMENT**

VAL(text)

Parses numeric text as a NUMBER.

```
? VAL("12.5") + 1
```

Result Prints 13.5.

BI-008 / DECIMAL**MAIN FRAGMENT**

DECIMAL(value)

Converts compatible input to exact DECIMAL.

```
? DECIMAL("19.95") + 0.05M
```

Result Prints 20; the result remains an exact DECIMAL.

BI-009 / TYPEOF**MAIN FRAGMENT**

TYPEOF(value)

Returns the runtime type name.

```
? TYPEOF({"id": 1})
```

Result Prints MAP.

BI-010 / DATE**MAIN FRAGMENT**

DATE()

Returns the current local date.

```
? TYPEOF(DATE())
```

Result Prints DATE.

BI-011 / DATETIME**MAIN FRAGMENT**

DATETIME()

Returns the current local date and time.

```
? TYPEOF(DATETIME())
```

Result Prints DATETIME.

BI-012 / JSONENCODE**MAIN FRAGMENT**

JSONENCODE(value)

Encodes supported Kokum values as JSON text. Native handles and cyclic graphs are not JSON values.

```
? JSONENCODE({"name": "Kokum", "active": .T.})
```

Result Prints a JSON object string.

BI-013 / JSONDECODE**MAIN FRAGMENT**

```
JSONDECODE(text)
```

Decodes JSON text into Kokum values.

```
LOCAL data AS JSON
data = JSONDECODE('{"count":2}')
? data["count"]
```

Result Prints 2.

BI-014 / MIN**MAIN FRAGMENT**

```
MIN(values AS ARRAY)
```

Returns the smallest element of one nonempty array. Elements must be mutually comparable.

```
? MIN([7, 3, 9])
```

Result Prints 3.

NOTE Pass one array, not multiple scalar arguments. Empty or incomparable arrays raise a catchable error.

BI-015 / MAX**MAIN FRAGMENT**

```
MAX(values AS ARRAY)
```

Returns the largest element of one nonempty array. Elements must be mutually comparable.

```
? MAX([7, 3, 9])
```

Result Prints 9.

NOTE Pass one array, not multiple scalar arguments. Empty or incomparable arrays raise a catchable error.

BI-016 / EVAL**MAIN FRAGMENT**

```
EVAL(block [, arguments ...])
```

Invokes a CODEBLOCK with optional arguments.

```
? EVAL({|x| x * 2}, 5)
```

Result Prints 10.

BI-017 / AADD**MAIN FRAGMENT**

```
AADD(array, value)
```

Appends one value to an array.

```
LOCAL a AS ARRAY
a = []
AADD(a, "first")
? a[1]
```

Result Prints first.

BI-018 / ASET**MAIN FRAGMENT**

```
ASET(array, index, value)
```

Replaces an existing one-based array element.

```
LOCAL a AS ARRAY
a = [10, 20]
ASET(a, 2, 25)
? a[2]
```

Result Prints 25.

BI-019 / AINS**MAIN FRAGMENT**

AINS(array, index, value)

Inserts a value at a one-based array position.

```
LOCAL a AS ARRAY
a = ["a", "c"]
AINS(a, 2, "b")
? JSONENCODE(a)
```

Result Prints ["a","b","c"].

BI-020 / ADEL**MAIN FRAGMENT**

ADEL(array, index)

Removes and returns a one-based array element.

```
LOCAL a AS ARRAY
a = ["a", "b"]
? ADEL(a, 1), LEN(a)
```

Result Prints a and 1.

BI-021 / MAPSET**MAIN FRAGMENT**

MAPSET(map, key, value)

Sets a map key and returns the stored value.

```
LOCAL m AS MAP
m = {}
MAPSET(m, "name", "Kokum")
? m["name"]
```

Result Prints Kokum.

BI-022 / HASKEY**MAIN FRAGMENT**

HASKEY(map, key)

Tests whether a map contains a key.

```
? HASKEY({"id": 7}, "id")
```

Result Prints .T.

BI-023 / MAPREMOVE**MAIN FRAGMENT**

MAPREMOVE(map, key)

Removes a map key and returns its previous value or NULL.

```
LOCAL m AS MAP
m = {"id": 7}
? MAPREMOVE(m, "id"), HASKEY(m, "id")
```

Result Prints 7 and .F.

BI-024 / KEYS**MAIN FRAGMENT**

KEYS(map)

Returns map keys in insertion order.

```
? JSONENCODE(KEYS({"a": 1, "b": 2}))
```

Result Prints ["a","b"].

BI-025 / VALUES

MAIN FRAGMENT

VALUES(map)

Returns map values in insertion order.

```
? JSONENCODE(VALUES({"a": 1, "b": 2}))
```

Result Prints [1,2].

Reflection class used by the next seven entries

```
CLASS Person
  PROPERTY Name AS STRING = "Anil"
  METHOD Greet AS STRING
    RETURN "Hello " + THIS.Name
  ENDMETHOD
ENDCLASS
```

Put this class above Main, then paste one of the following reflection fragments inside Main.

BI-026 / CLASSNAME

MAIN FRAGMENT + SHARED DECLARATION · SET-REFLECTION

CLASSNAME(object)

Returns the effective class name of an object.

```
LOCAL p
p = NEW Person()
? CLASSNAME(p)
```

Result Prints Person.

NOTE Use the Person class from SET-REFLECTION.

BI-027 / PROPERTIES

MAIN FRAGMENT + SHARED DECLARATION · SET-REFLECTION

PROPERTIES(object)

Returns the public property names of an object.

```
LOCAL p
p = NEW Person()
? JSONENCODE(PROPERTIES(p))
```

Result Prints an array containing Name.

NOTE Use the Person class from SET-REFLECTION.

BI-028 / METHODS

MAIN FRAGMENT + SHARED DECLARATION · SET-REFLECTION

METHODS(object)

Returns the public method names of an object.

```
LOCAL p
p = NEW Person()
? JSONENCODE(METHODS(p))
```

Result Prints an array containing Greet.

NOTE Use the Person class from SET-REFLECTION.

BI-029 / HASPROPERTY**MAIN FRAGMENT + SHARED DECLARATION · SET-REFLECTION**

HASPROPERTY(object, name)

Tests whether an object exposes a property.

```
LOCAL p
p = NEW Person()
? HASPROPERTY(p, "Name")
```

Result Prints .T.

NOTE Use the Person class from SET-REFLECTION.

BI-030 / HASMETHOD**MAIN FRAGMENT + SHARED DECLARATION · SET-REFLECTION**

HASMETHOD(object, name)

Tests whether an object exposes a method.

```
LOCAL p
p = NEW Person()
? HASMETHOD(p, "Greet")
```

Result Prints .T.

NOTE Use the Person class from SET-REFLECTION.

BI-031 / GETPROPERTY**MAIN FRAGMENT + SHARED DECLARATION · SET-REFLECTION**

GETPROPERTY(object, name)

Reads an object property by name.

```
LOCAL p
p = NEW Person()
? GETPROPERTY(p, "Name")
```

Result Prints Anil.

NOTE Use the Person class from SET-REFLECTION.

BI-032 / SETPROPERTY**MAIN FRAGMENT + SHARED DECLARATION · SET-REFLECTION**

SETPROPERTY(object, name, value)

Writes an object property by name.

```
LOCAL p
p = NEW Person()
SETPROPERTY(p, "Name", "Kokum")
? p.Name
```

Result Prints Kokum.

NOTE Use the Person class from SET-REFLECTION.

BI-033 / GCCOLLECT**MAIN FRAGMENT**

GCCOLLECT()

Runs explicit cycle collection and returns the number of reclaimed nodes.

```
LOCAL cycle
cycle = []
AADD(cycle, cycle)
cycle = NULL
? TYPEOF(GCCOLLECT())
```

Result Prints INTEGER. The reclaimed-node count is runtime-dependent.

NOTE Explicit collection is diagnostic; do not depend on a particular reclamation count.

BI-034 / GCSTATS**MAIN FRAGMENT**`GCSTATS()`

Returns runtime memory and garbage-collection diagnostics.

```
? TYPEOF(GCSTATS())
```

Result Prints MAP. Inspect the map for runtime-specific counters.

07 / REFERENCE

Mathematics and array statistics

Use scalar functions for individual numbers and array functions for collections.

SOURCE BASIS [S3, S6] Scalar mathematics and array-statistics contracts.

MIN and MAX take exactly one nonempty ARRAY, not several scalar arguments. Statistical functions skip NULL elements, but reject other unsupported element types. SUM of an empty numeric collection is zero; other statistics may return NULL when insufficient numeric values are present. VARIANCE and STDDEV default to population mode; .T. selects sample mode.

BI-039 / SIGN

MAIN FRAGMENT

SIGN(number)

Returns -1, 0, or 1 according to a number's sign.

```
? SIGN(-5), SIGN(0), SIGN(8)
```

Result Prints -1, 0, and 1.

BI-040 / ROUND

MAIN FRAGMENT

ROUND(number [, places])

Rounds a number, optionally to a decimal-place count.

```
? ROUND(12.345, 2)
```

Result Prints 12.35.

BI-041 / FLOOR

MAIN FRAGMENT

FLOOR(number)

Returns the greatest integer not above the input.

```
? FLOOR(3.9)
```

Result Prints 3.

BI-042 / CEIL

MAIN FRAGMENT

CEIL(number)

Returns the least integer not below the input.

```
? CEIL(3.1)
```

Result Prints 4.

BI-043 / CEILING

MAIN FRAGMENT

CEILING(number)

Alias of CEIL.

```
? CEILING(3.1)
```

Result Prints 4.

BI-044 / TRUNC**MAIN FRAGMENT**

```
TRUNC(number [, places])
```

Truncates a number, optionally to a decimal-place count.

```
? TRUNC(12.987, 2)
```

Result Prints 12.98.

BI-045 / SQRT**MAIN FRAGMENT**

```
SQRT(number)
```

Returns the square root.

```
? SQRT(81)
```

Result Prints 9.

BI-046 / POW**MAIN FRAGMENT**

```
POW(base, exponent)
```

Raises a number to a power.

```
? POW(2, 8)
```

Result Prints 256.

BI-047 / EXP**MAIN FRAGMENT**

```
EXP(number)
```

Returns e raised to the supplied power.

```
? EXP(0)
```

Result Prints 1.

BI-048 / LOG**MAIN FRAGMENT**

```
LOG(number)
```

Returns the natural logarithm.

```
? ROUND(LOG(EXP(1)), 6)
```

Result Prints 1.

BI-049 / LOG10**MAIN FRAGMENT**

```
LOG10(number)
```

Returns the base-10 logarithm.

```
? LOG10(1000)
```

Result Prints 3.

BI-050 / MOD**MAIN FRAGMENT**

```
MOD(dividend, divisor)
```

Returns the numeric remainder.

```
? MOD(17, 5)
```

Result Prints 2.

BI-051 / CLAMP**MAIN FRAGMENT**

CLAMP(value, minimum, maximum)

Constrains a value to an inclusive minimum/maximum range.

```
? CLAMP(15, 0, 10)
```

Result Prints 10.

BI-052 / SUM**MAIN FRAGMENT**

SUM(values)

Returns the sum of numeric array values while ignoring NULL.

```
? SUM([10, NULL, 20, 30])
```

Result Prints 60.

BI-053 / MEAN**MAIN FRAGMENT**

MEAN(values)

Returns the arithmetic mean of numeric array values.

```
? MEAN([10, 20, 30])
```

Result Prints 20.

BI-054 / AVERAGE**MAIN FRAGMENT**

AVERAGE(values)

Alias of MEAN.

```
? AVERAGE([10, 20, 30])
```

Result Prints 20.

BI-055 / MEDIAN**MAIN FRAGMENT**

MEDIAN(values)

Returns the middle value or midpoint after sorting a copy.

```
? MEDIAN([4, 1, 3, 2])
```

Result Prints 2.5.

BI-056 / VARIANCE**MAIN FRAGMENT**

VARIANCE(values [, sample])

Returns population variance, or sample variance when sample is .T..

```
? VARIANCE([2, 4, 4, 4, 5, 5, 7, 9])
```

Result Prints 4.

BI-057 / STDDEV**MAIN FRAGMENT**

STDDEV(values [, sample])

Returns the square root of the corresponding variance.

```
? STDDEV([2, 4, 4, 4, 5, 5, 7, 9])
```

Result Prints 2.

BI-058 / PERCENTILE**MAIN FRAGMENT**

PERCENTILE(values, percentile)

Returns an R-7 linearly interpolated percentile from 0 through 100.

```
? PERCENTILE([1, 2, 3, 4], 25)
```

Result Prints 1.75.

BI-059 / COUNTNUM**MAIN FRAGMENT**

COUNTNUM(values)

Counts numeric, non-NULL elements in an array.

```
? COUNTNUM([10, NULL, 20])
```

Result Prints 2.

BI-060 / RANGE**MAIN FRAGMENT**

RANGE(values)

Returns maximum minus minimum for numeric array values.

```
? RANGE([10, 30, 20])
```

Result Prints 20.

08 / REFERENCE

Files, regular expressions and natural patterns

Start with owned file handles, then choose a direct regex or a readable natural-pattern statement.

SOURCE BASIS [S7–S8] FILE I/O, regex runtime and natural-pattern implementations.

File examples write only named lab files in the working directory. READLINE returns NULL at end-of-file; an empty STRING is a genuine blank line. There is no EOF() or SEEK() built-in in this catalogue.

BI-061 / OPEN

MAIN FRAGMENT

OPEN(path, mode)

Opens a UTF-8 text file and returns a FILE resource.

```
LOCAL file AS FILE
file = OPEN("sample.txt", "w")
? TYPEOF(file)
CLOSEFILE(file)
```

Result Prints FILE and creates/truncates sample.txt.

BI-062 / READ

MAIN FRAGMENT

READ(file [, byteCount])

Reads the remainder or a bounded byte count as UTF-8 text.

```
LOCAL file AS FILE
file = OPEN("sample.txt", "w")
WRITE(file, "Alpha\nBeta\n")
CLOSEFILE(file)
file = OPEN("sample.txt", "r")
? READ(file, 5)
CLOSEFILE(file)
```

Result Prints Alpha. READ counts bytes and advances the file position.

BI-063 / WRITE

MAIN FRAGMENT

WRITE(file, text)

Writes UTF-8 text at the current stream position and returns bytes written.

```
LOCAL file AS FILE
file = OPEN("sample.txt", "w")
? WRITE(file, "Kokum\n")
CLOSEFILE(file)
```

Result Writes Kokum plus newline and prints the byte count.

BI-064 / APPEND

MAIN FRAGMENT

APPEND(file, text)

Writes UTF-8 text at end of file and returns bytes written.

```
LOCAL file AS FILE
file = OPEN("sample.txt", "a")
APPEND(file, "Second line\n")
CLOSEFILE(file)
```

Result Adds Second line without replacing existing data.

BI-065 / CLOSEFILE

MAIN FRAGMENT

CLOSEFILE(file)

Closes a FILE resource idempotently; NULL is safe.

```

LOCAL file AS FILE
file = OPEN("close-demo.txt", "w")
CLOSEFILE(file)
CLOSEFILE(file)
? "closed"

```

Result Prints closed. Repeated closure is safe.

BI-066 / READLINE

MAIN FRAGMENT

READLINE(file)

Reads one line and advances the file pointer; returns NULL at EOF.

```

LOCAL file AS FILE
LOCAL line
file = OPEN("lines.txt", "w")
WRITE(file, "Alpha\n\nBeta\n")
CLOSEFILE(file)
file = OPEN("lines.txt", "r")
DO WHILE .T.
    line = READLINE(file)
    IF line = NULL
        EXIT
    ENDIF
    ? line
ENDDO
CLOSEFILE(file)

```

Result Prints Alpha, a blank line, then Beta. NULL signals EOF; an empty string is a real blank line.

NOTE No EOF() or SEEK() built-in exists in this catalogue. A read loop stops on READLINE() = NULL.

Direct regex built-ins

Pattern arguments precede the input. REGEXFIND returns a STRING or NULL and REGEXFINDALL returns an ARRAY of strings. Neither returns capture-detail MAPs. Those maps belong to the natural-pattern WITH DETAILS form.

BI-097 / REGEXCOMPILE

MAIN FRAGMENT

REGEXCOMPILE(pattern [, flags])

Compiles a reusable REGEX resource. PATTERN is its annotation alias.

```

LOCAL rx AS REGEX
rx = REGEXCOMPILE("[A-Z]+", "i")
? rx.IsMatch("kokum")
rx.Close()

```

Result Prints .T.

BI-098 / REGEXMATCH

MAIN FRAGMENT

REGEXMATCH(patternOrRegex, textOrFile [, flags])

Tests whether any match exists; anchors are needed for whole-string validation.

```

? REGEXMATCH("^[A-Z]{2}[0-9]{2}$", "AB12")

```

Result Prints .T.

BI-099 / REGEXFIND

MAIN FRAGMENT

```
REGEXFIND(patternOrRegex, textOrFile [, flags])
```

Returns the first matched STRING, or NULL. Detailed capture maps are obtained through natural FIND ... WITH DETAILS, not this function.

```
LOCAL hit
hit = REGEXFIND("[0-9]+", "ID 42")
IF hit != NULL
    ? JSONENCODE(hit)
ENDIF
```

Result Prints the JSON string "42".

NOTE The result is not a match MAP. Test NULL before using it.

BI-100 / REGEXFINDALL

MAIN FRAGMENT

```
REGEXFINDALL(patternOrRegex, textOrFile [, flags])
```

Returns an ARRAY of matched strings in non-overlapping order, not match maps.

```
? LEN(REGEXFINDALL("[0-9]+", "A1 B22"))
```

Result Prints 2.

BI-101 / REGEXREPLACE

MAIN FRAGMENT

```
REGEXREPLACE(patternOrRegex, textOrFile, replacement [, flags])
```

Returns replacement text; does not modify an input FILE.

```
? REGEXREPLACE("[0-9]+", "A12 B34", "#")
```

Result Prints A# B#.

BI-102 / REGEXSPLIT

MAIN FRAGMENT

```
REGEXSPLIT(patternOrRegex, textOrFile [, flags])
```

Splits the subject around regular-expression delimiters.

```
? JSONENCODE(REGEXSPLIT("[,;]", "one,two;three"))
```

Result Prints ["one","two","three"].

BI-103 / REGEXESCAPE

MAIN FRAGMENT

```
REGEXESCAPE(text)
```

Escapes metacharacters so input is matched literally.

```
LOCAL pattern AS STRING
pattern = "^" + REGEXESCAPE("a+b") + "$"
? REGEXMATCH(pattern, "a+b")
```

Result Prints .T.

Compiled REGEX methods

Each method fragment prepares a REGEX handle and closes it after use. Pattern(), Flags() and IsClosed() are methods with parentheses, not properties.

API-103 / REGEX.IsMatch**MAIN FRAGMENT**

```
IsMatch(textOrFile)
```

Tests for a match.

```
LOCAL rx AS REGEX
rx = REGEXCOMPILE("[0-9]+")
? rx.IsMatch("A12")
rx.Close()
```

Result Prints .T.

API-104 / REGEX.Test**MAIN FRAGMENT**

```
Test(textOrFile)
```

Alias of IsMatch.

```
LOCAL rx AS REGEX
rx = REGEXCOMPILE("[0-9]+")
? rx.Test("A12")
rx.Close()
```

Result Prints .T.

API-105 / REGEX.Find**MAIN FRAGMENT**

```
Find(textOrFile)
```

Returns the first matched STRING, or NULL.

```
LOCAL rx AS REGEX
rx = REGEXCOMPILE("[0-9]+")
? JSONENCODE(rx.Find("A12"))
rx.Close()
```

Result Prints the JSON string "12".

API-106 / REGEX.FindAll**MAIN FRAGMENT**

```
FindAll(textOrFile)
```

Returns all non-overlapping matched strings.

```
LOCAL rx AS REGEX
rx = REGEXCOMPILE("[0-9]+")
? LEN(rx.FindAll("A12 B3"))
rx.Close()
```

Result Prints 2.

API-107 / REGEX.Replace**MAIN FRAGMENT**

```
Replace(textOrFile, replacement)
```

Replaces matches and returns new text.

```
LOCAL rx AS REGEX
rx = REGEXCOMPILE("[0-9]+")
? rx.Replace("A12 B3", "#")
rx.Close()
```

Result Prints A# B#.

API-108 / REGEX.Split

MAIN FRAGMENT

Split(textOrFile)

Splits around this compiled pattern.

```

LOCAL rx AS REGEX
rx = REGEXCOMPILE("[0-9]+")
? JSONENCODE(rx.Split("A12B3C"))
rx.Close()

```

Result Prints ["A","B","C"].

API-109 / REGEX.Pattern

MAIN FRAGMENT

Pattern()

Returns the source pattern string.

```

LOCAL rx AS REGEX
rx = REGEXCOMPILE("[0-9]+")
? rx.Pattern()
rx.Close()

```

Result Prints [0-9]+.

API-110 / REGEX.Flags

MAIN FRAGMENT

Flags()

Returns the normalized flag string.

```

LOCAL rx AS REGEX
rx = REGEXCOMPILE("[0-9]+")
? TYPEOF(rx.Flags())
rx.Close()

```

Result Prints STRING.

API-111 / REGEX.IsClosed

MAIN FRAGMENT

IsClosed()

Tests whether the compiled resource is closed.

```

LOCAL rx AS REGEX
rx = REGEXCOMPILE("[0-9]+")
? rx.IsClosed()
rx.Close()

```

Result Prints .F. before Close().

API-112 / REGEX.Close

MAIN FRAGMENT

Close()

Releases compiled pattern resources.

```

LOCAL rx AS REGEX
rx = REGEXCOMPILE("[0-9]+")
rx.Close()
? rx.IsClosed()

```

Result Prints .T.

Flags and resource limits

Flag	Meaning
u	Unicode-property mode; default.

Flag	Meaning
a	ASCII character-property mode while retaining UTF-8 text; cannot be combined with u.
i / m / s	Case-insensitive / multiline anchors / dot matches newline.
x	Extended pattern whitespace/comments.
U	Invert default greediness.
n	Disable ordinary unnamed captures.

The documented limits bound pattern size (64 KiB), subject size (64 MiB) and result count (100,000). They are not a guarantee that arbitrary patterns are inexpensive. FILE input starts at the current file position and does not implicitly close your handle.

Natural-pattern commands

FIND, REPLACE, SPLIT, VALIDATE and ITERATE have their own contextual grammar. The examples create their own inputs. Transformations produce results; they do not silently overwrite the input file. WITH DETAILS exposes position/capture maps; selected fields include Value, Source, Line, Column, Start, End, Length, UnitIndex, MatchIndex, Groups and NamedGroups. Byte offsets and line/column locations use different bases; do not treat them as normal array indexes.

PAT-001 / FIND ... LINES

MAIN FRAGMENT

```
FIND FROM source selector LINES WHERE condition INTO target
```

Selects source lines, then filters them. Line terminators are omitted.

```
LOCAL rows AS ARRAY
FIND FROM TEXT "Apple\nBerry\nApricot" ALL LINES ;
  WHERE STARTS WITH "A" INTO rows
? JSONENCODE(rows)
```

Result Prints ["Apple","Apricot"].

PAT-002 / FIND ... WORDS

MAIN FRAGMENT

```
FIND FROM source selector WORDS WHERE condition INTO target
```

Searches Unicode-aware word units instead of entire lines.

```
LOCAL words AS ARRAY
FIND FROM TEXT "red blue red" ALL WORDS ;
  WHERE EQUALS "red" INTO words
? LEN(words)
```

Result Prints 2.

PAT-003 / FIND ... COUNT TO

MAIN FRAGMENT

```
FIND FROM source ... WHERE condition COUNT TO count
```

Counts results without retaining every result value.

```
LOCAL count AS INTEGER
FIND FROM TEXT "OK\nERROR\nERROR" ALL LINES ;
  WHERE CONTAINS "ERROR" COUNT TO count
? count
```

Result Prints 2.

PAT-004 / FIND selectors

MAIN FRAGMENT

FIRST n | LAST n | ALTERNATE | EXCEPT FIRST/LAST

For LINES/WORDS, selection refers to source units before filtering. LIMIT caps successful results.

```

LOCAL rows AS ARRAY
FIND FROM TEXT "A\nB\nC\nD" ALTERNATE LINES ;
    STARTING WITH 2 WHERE IS NOT BLANK INTO rows
? JSONENCODE(rows)

```

Result Prints ["B","D"].

PAT-005 / FIND ... WITH DETAILS

MAIN FRAGMENT

FIND ... MATCHES WHERE MATCHES REGEX pattern WITH DETAILS INTO target

Returns match metadata including captures, positions and source.

```

LOCAL hits AS ARRAY
FIND FROM TEXT "Order AB-12" ALL MATCHES ;
    WHERE MATCHES REGEX "([A-Z]+)-([0-9]+)" ;
    WITH DETAILS INTO hits
? hits[1]["Value"]

```

Result Prints AB-12.

PAT-006 / REPLACE FROM

MAIN FRAGMENT

REPLACE FROM source ... WHERE pattern WITH replacement INTO target

Returns transformed text. It never overwrites an input file implicitly.

```

LOCAL output AS STRING
REPLACE FROM TEXT "AA12 BB340" ALL MATCHES
    WHERE CAPTURE EXACTLY 2 LETTERS AS code ;
    FOLLOWED BY CAPTURE ONE OR MORE DIGITS AS number
    WITH "${code}.$2"
    INTO output
? output

```

Result Prints AA:12 BB:340.

PAT-007 / SPLIT FROM

MAIN FRAGMENT

SPLIT FROM source WHERE pattern [KEEP DELIMITERS] INTO target

Splits into an ARRAY. Empty fields are retained; LIMIT bounds delimiters consumed.

```

LOCAL parts AS ARRAY
SPLIT FROM TEXT "one,two;three"
    WHERE ONE OF ",;"
    KEEP DELIMITERS
    INTO parts
? JSONENCODE(parts)

```

Result Prints ["one",",","two",",","three"].

PAT-008 / VALIDATE FROM

MAIN FRAGMENT

VALIDATE FROM source WHERE condition INTO logicalTarget

Natural sequences and raw regex use whole-source validation. IS BLANK means empty, not whitespace-trimmed.

```

LOCAL valid AS LOGICAL
VALIDATE FROM TEXT "AA12"
    WHERE EXACTLY 2 LETTERS FOLLOWED BY EXACTLY 2 DIGITS
    INTO valid
? valid

```

Result Prints .T.

PAT-009 / ITERATE FROM / ENDITERATE**MAIN FRAGMENT**

```
ITERATE FROM source ... AS item [WITH INDEX n] ... ENDITERATE
```

Iterates ordinary statements over detailed match maps. The iteration index is one-based.

```
LOCAL item AS MAP
LOCAL position AS INTEGER
ITERATE FROM TEXT "A12 B3" ALL MATCHES
  WHERE ONE OR MORE DIGITS
  AS item WITH INDEX position
  ? position, item["Value"]
ENDITERATE item
```

Result Prints 1 12, then 2 3.

PAT-010 / Natural FILE source**MAIN FRAGMENT**

```
FIND FROM FILE handle ...
```

Searches from an existing FILE position; the caller still owns the handle. A path source is opened and closed internally.

```
LOCAL f AS FILE
LOCAL rows AS ARRAY
f = OPEN("find-demo.txt", "w")
WRITE(f, "Alpha\nBeta\nApricot\n")
CLOSEFILE(f)
f = OPEN("find-demo.txt", "r")
FIND FROM FILE f ALL LINES ;
  WHERE STARTS WITH "A" INTO rows
CLOSEFILE(f)
? LEN(rows)
```

Result Prints 2. Creates find-demo.txt in the working directory.

09 / REFERENCE

Forms, signals, slots and control APIs

A GUI slot is ordinary Kokum code, but its receivers are supplied by the form runtime.

SOURCE BASIS [S9] Form schema, slot binding and native web-form bridge.

The standard slot context

Create a Web GUI project in the Designer. Add a Button named button1 and a Label named statusBar1; bind the button's clicked event to button1_clicked. The form's slotsModule must match the MODULE name in slots.kokum. Keep the three parameter names sender, event and form; export the handler. Do not redeclare event as a LOCAL inside that slot.

```
MODULE referencedemo.slots VERSION "1.0.0"
EXPORT PROCEDURE button1_clicked

PROCEDURE button1_clicked(sender, event, form)
    form.statusBar1.Text = "Clicked"
ENDPROC
```

Use your actual generated module name. Sender is the source control, event carries the event data, and form provides control access by stable ID. Generated input values and database/resource handles are placed in marked PUBLIC blocks. Never duplicate those declarations or put your own variables inside those blocks.

LANG-057 / Event procedure

GUI PROJECT

```
PROCEDURE control_event(sender, event, form)
```

Implements a Designer-connected slot.

```
EXPORT PROCEDURE button1_clicked

PROCEDURE button1_clicked(sender, event, form)
    form.statusBar1.Text = "Clicked"
ENDPROC
```

Result Clicking button1 changes the status text.

NOTE The module name and explicit export must match the form binding. Parameters are named sender, event, form, in that order.

LANG-058 / Form control property write

GUI PROJECT

```
form.controlId.Property = value
```

Changes a runtime-mutable control property.

```
form.txtName.Text = "Anil"
form.txtName.Enabled = .T.
form.statusBar1.Text = "Ready"
```

Result The name field and status label update.

LANG-059 / Automatic widget variable

GUI PROJECT

```
PUBLIC controlId AS mappedType
```

Reads or writes the value synchronized for an input/data control.

```
PUBLIC txtName AS STRING

PROCEDURE btnRead_clicked(sender, event, form)
    ? txtName
    txtName = "Updated"
ENDPROC
```

Result The current text is printed and the TextBox changes to Updated.

NOTE Place your own PUBLIC declarations outside automatic widget/database blocks. Preserve generated declarations; the baseline includes the PUBLIC-preservation IDE fix.

Form lifecycle prerequisites

The lifecycle examples require registered forms named `CustomerView` and `SettingsWindow`, and a designer-owned subform instance named `CustomerPane` where shown. Create those names in the project first. They are not inferred from strings or variables, and a missing form cannot be made runnable by declaring a `PUBLIC` variable. Use a separate launcher form for tests that hide or close a view.

LANG-053 / SHOW

GUI PROJECT

`SHOW FormName`

Displays or activates a compiled window or subform.

```
PROCEDURE btnOpen_clicked(sender, event, form)
    SHOW CustomerView
ENDPROC
```

Result `CustomerView` becomes visible.

LANG-054 / HIDE

GUI PROJECT · SET-FORMS

`HIDE FormName`

Hides a form without disposing it.

```
PROCEDURE btnHide_clicked(sender, event, form)
    HIDE CustomerView
ENDPROC
```

Result `CustomerView` is hidden and can be shown again.

LANG-055 / CLOSE

GUI PROJECT · SET-FORMS

`CLOSE FormName`

Requests that a window close.

```
PROCEDURE btnClose_clicked(sender, event, form)
    CLOSE CustomerView
ENDPROC
```

Result `CustomerView` closes according to its lifecycle.

NOTE `CLOSE` concerns forms; use `CLOSEFILE(handle)` for an open text file.

LANG-056 / DISPOSE

GUI PROJECT · SET-FORMS

`DISPOSE FormName`

Disposes an owned subform instance. A later `SHOW` recreates the instance according to its form definition.

```
PROCEDURE btnDispose_clicked(sender, event, form)
    DISPOSE CustomerPane
ENDPROC
```

Result `CustomerPane` is disposed. `SHOW` can create a fresh instance.

NOTE Use `DISPOSE` for subforms; do not treat it as a generic object destructor.

BI-035 / SHOWFORM

GUI PROJECT · SET-FORMS

`SHOWFORM(formName)`

Shows a compiled form by identifier.

```
SHOWFORM("CustomerView")
```

Result `CustomerView` becomes visible.

BI-036 / HIDEFORM**GUI PROJECT · SET-FORMS**

HIDEFORM(formName)

Hides a compiled form by identifier.

```
HIDEFORM("CustomerView")
```

Result CustomerView becomes hidden.**BI-037 / CLOSEFORM****GUI PROJECT · SET-FORMS**

CLOSEFORM(formName)

Closes a compiled form by identifier.

```
CLOSEFORM("CustomerView")
```

Result CustomerView closes.**Control lab prerequisites**

ID	Designer type / initial state
combo1	ComboBox. Before each item example, call SetItems(["One","Two","Three"]).
scroll1 / panelDetails	ScrollView 200×100 containing panelDetails at y=300, height 150.
splitMain	Splitter with two Panel children; initial position 200.
tree1	TreeView. Its slot Items property is an ARRAY of indented line strings.
table1	TableView. For update/removal/cell examples, create id and name columns, then add {"id":1,"name":"Anil"}.
grid1	PropertyGrid. Before value/removal examples, add the General/name property shown below.

```
form.table1.ClearRows()
form.table1.ClearColumns()
form.table1.AddColumn("id", "ID")
form.table1.AddColumn("name", "Name")
form.table1.AddRow({"id":1,"name":"Anil"})

form.grid1.Clear()
form.grid1.AddProperty( ;
    "General", "name", "Name", "Anil", "string", .F.)
```

Restart the form or restore the stated initial state before each independent example. For AddColumn/AddRow demonstrations, start with empty columns/rows rather than adding duplicate column IDs. Item-oriented ComboBox insertion/removal is one-based; SelectedIndex and TableView indexes are zero-based.

Important browser/native distinction

The shared browser contract advertises ScrollTo, ScrollBy, EnsureVisible, SetSplitPosition, ResetSplit and six TreeView methods. Native slot dispatch in this baseline does not implement those 11 methods on the control MAP. They are listed here for identification, with tested property-based Kokum alternatives. Expand/Collapse remain user interactions rather than programmable Kokum methods. This book does not change the runtime.

WID-001 / ComboBox.SetItems**GUI CONTROL: COMBOBOX · SET-CONTROLS**

form.combo1.SetItems(items AS ARRAY)

Replaces all choices.

```
? form.combo1.SetItems(["One","Two","Three"])
```

Result Prints 3 and displays three choices.

WID-002 / ComboBox.AddItem**GUI CONTROL: COMBOBOX · SET-CONTROLS**`form.combo1.AddItem(value AS STRING)`

Appends one choice.

```
? form.combo1.AddItem("Four")
```

Result Prints the new item count.**WID-003 / ComboBox.InsertItem****GUI CONTROL: COMBOBOX · SET-CONTROLS**`form.combo1.InsertItem(index AS INTEGER, value AS STRING)`

Inserts at a one-based index.

```
form.combo1.InsertItem(2, "One and Half")
```

Result The new item is second.**WID-004 / ComboBox.RemoveItem****GUI CONTROL: COMBOBOX · SET-CONTROLS**`form.combo1.RemoveItem(value AS STRING)`

Removes the first matching choice.

```
? form.combo1.RemoveItem("Two")
```

Result Prints .T. when Two existed.**WID-005 / ComboBox.RemoveItemAt****GUI CONTROL: COMBOBOX · SET-CONTROLS**`form.combo1.RemoveItemAt(index AS INTEGER)`

Removes and returns a one-based item.

```
? form.combo1.RemoveItemAt(1)
```

Result Prints the removed first item.**WID-006 / ComboBox.ClearItems****GUI CONTROL: COMBOBOX · SET-CONTROLS**`form.combo1.ClearItems()`

Removes all choices and selection.

```
form.combo1.ClearItems()
? form.combo1.ItemCount
```

Result Prints 0.**WID-007 / ComboBox.ContainsItem****GUI CONTROL: COMBOBOX · SET-CONTROLS**`form.combo1.ContainsItem(value AS STRING)`

Tests whether a choice exists.

```
? form.combo1.ContainsItem("Three")
```

Result Prints .T. or .F.

WID-008 / ScrollView.ScrollTo

GUI — BROWSER API DISTINCTION · SET-CONTROLS

Property-based Kokum example (not a native ScrollTo call)

The named method belongs to the browser control API and is not callable on the Kokum slot MAP in this baseline.

```
form.scroll1.ScrollX = 0
form.scroll1.ScrollY = 300
```

Result Sets the scroll position through the supported properties.

NOTE Do not paste the browser method name into a Kokum slot. The shown property assignment is the supported alternative.

WID-009 / ScrollView.ScrollBy

GUI — BROWSER API DISTINCTION · SET-CONTROLS

Property-based Kokum example (not a native ScrollBy call)

The named method belongs to the browser control API and is not callable on the Kokum slot MAP in this baseline.

```
form.scroll1.ScrollY = form.scroll1.ScrollY + 50
```

Result Moves down 50 logical pixels through the scroll property.

NOTE Do not paste the browser method name into a Kokum slot. The shown property assignment is the supported alternative.

WID-010 / ScrollView.EnsureVisible

GUI — BROWSER API DISTINCTION · SET-CONTROLS

Property-based Kokum example (not a native EnsureVisible call)

The named method belongs to the browser control API and is not callable on the Kokum slot MAP in this baseline.

```
form.scroll1.ScrollY = 300
```

Result Scrolls to the known top coordinate of panelDetails in this lab; not a general EnsureVisible replacement.

NOTE Do not paste the browser method name into a Kokum slot. The shown property assignment is the supported alternative.

WID-011 / Splitter.SetSplitPosition

GUI — BROWSER API DISTINCTION · SET-CONTROLS

Property-based Kokum example (not a native SetSplitPosition call)

The named method belongs to the browser control API and is not callable on the Kokum slot MAP in this baseline.

```
form.splitMain.Position = 240
```

Result Assigns the canonical position property; the sample first pane is 240 pixels.

NOTE Do not paste the browser method name into a Kokum slot. The shown property assignment is the supported alternative.

WID-012 / Splitter.ResetSplit

GUI — BROWSER API DISTINCTION · SET-CONTROLS

Property-based Kokum example (not a native ResetSplit call)

The named method belongs to the browser control API and is not callable on the Kokum slot MAP in this baseline.

```
form.splitMain.Position = 200
```

Result Restores this lab's chosen initial position explicitly.

NOTE Do not paste the browser method name into a Kokum slot. The shown property assignment is the supported alternative.

WID-013 / TreeView.AddItem

GUI — BROWSER API DISTINCTION · SET-CONTROLS

Property-based Kokum example (not a native AddItem call)

The named method belongs to the browser control API and is not callable on the Kokum slot MAP in this baseline.

```
form.tree1.Items = ["Root", " Child A"]
```

Result Replaces the item text with one root and an indented child.

NOTE Do not paste the browser method name into a Kokum slot. The shown property assignment is the supported alternative. In a Kokum slot, Items is an ARRAY of indented line strings; the designer stores newline-delimited text.

WID-014 / TreeView.RemoveItem

GUI — BROWSER API DISTINCTION · SET-CONTROLS

Property-based Kokum example (not a native RemoveItem call)

The named method belongs to the browser control API and is not callable on the Kokum slot MAP in this baseline.

```
form.tree1.Items = ["Root"]
```

Result Replaces the tree data with the remaining root; no incremental RemoveItem call is issued.

NOTE Do not paste the browser method name into a Kokum slot. The shown property assignment is the supported alternative. In a Kokum slot, Items is an ARRAY of indented line strings; the designer stores newline-delimited text.

WID-015 / TreeView.Clear

GUI — BROWSER API DISTINCTION · SET-CONTROLS

Property-based Kokum example (not a native Clear call)

The named method belongs to the browser control API and is not callable on the Kokum slot MAP in this baseline.

```
form.tree1.Items = []
```

Result Clears the complete tree through its supported data property.

NOTE Do not paste the browser method name into a Kokum slot. The shown property assignment is the supported alternative. In a Kokum slot, Items is an ARRAY of indented line strings; the designer stores newline-delimited text.

WID-016 / TreeView.Expand

GUI — BROWSER API DISTINCTION · SET-CONTROLS

Property-based Kokum example (not a native Expand call)

The named method belongs to the browser control API and is not callable on the Kokum slot MAP in this baseline.

```
form.tree1.Items = ["Root", " Child A"]
```

Result Loads an expandable node; expand it using the runtime's disclosure control. This code does not call Expand.

NOTE Verified baseline distinction: invoking the named method from a Kokum slot raises TLR3014. Use the property example above; expansion/collapse remains a user interaction. In a Kokum slot, Items is an ARRAY of indented line strings; the designer stores newline-delimited text.

WID-017 / TreeView.Collapse

GUI — BROWSER API DISTINCTION · SET-CONTROLS

Property-based Kokum example (not a native Collapse call)

The named method belongs to the browser control API and is not callable on the Kokum slot MAP in this baseline.

```
form.tree1.Items = ["Root", " Child A"]
```

Result Loads the tree; collapse it using the runtime's disclosure control. No Kokum Collapse method is available here.

NOTE Verified baseline distinction: invoking the named method from a Kokum slot raises TLR3014. Use the property example above; expansion/collapse remains a user interaction. In a Kokum slot, Items is an ARRAY of indented line strings; the designer stores newline-delimited text.

WID-018 / TreeView.Select

GUI — BROWSER API DISTINCTION · SET-CONTROLS

Property-based Kokum example (not a native Select call)

The named method belongs to the browser control API and is not callable on the Kokum slot MAP in this baseline.

```
form.tree1.Items = ["Root", " Child A"]
form.tree1.SelectedPath = "Root/Child A"
```

Result Uses the supported text-path selection property.

NOTE Do not paste the browser method name into a Kokum slot. The shown property assignment is the supported alternative. In a Kokum slot, Items is an ARRAY of indented line strings; the designer stores newline-delimited text.

WID-019 / TableView.AddColumn**GUI CONTROL: TABLEVIEW · SET-CONTROLS**

```
form.table1.AddColumn(id [, caption [, width [, alignment]]])
```

Adds a uniquely identified column.

```
? form.table1.AddColumn("id", "ID", 80, "right")
```

Result Prints the resulting column count.

WID-020 / TableView.AddRow**GUI CONTROL: TABLEVIEW · SET-CONTROLS**

```
form.table1.AddRow(row MAP|ARRAY or cell values...)
```

Appends a row and returns its zero-based index.

```
form.table1.AddColumn("id", "ID")
form.table1.AddColumn("name", "Name")
? form.table1.AddRow({"id":1, "name":"Anil"})
```

Result Prints 0 for the first row.

WID-021 / TableView.RemoveColumn**GUI CONTROL: TABLEVIEW · SET-CONTROLS**

```
form.table1.RemoveColumn(indexOrName)
```

Removes a zero-based column or named column.

```
? form.table1.RemoveColumn("id")
```

Result Prints .T. when id existed.

WID-022 / TableView.RemoveRow**GUI CONTROL: TABLEVIEW · SET-CONTROLS**

```
form.table1.RemoveRow(zeroBasedIndex)
```

Removes a row by zero-based index.

```
? form.table1.RemoveRow(0)
```

Result Prints .T. when the first row existed.

WID-023 / TableView.UpdateRow**GUI CONTROL: TABLEVIEW · SET-CONTROLS**

```
form.table1.UpdateRow(zeroBasedIndex, row)
```

Replaces/merges a row.

```
? form.table1.UpdateRow(0, {"id":1, "name":"Updated"})
```

Result Prints .T. and updates the first row.

WID-024 / TableView.ClearRows**GUI CONTROL: TABLEVIEW · SET-CONTROLS**

```
form.table1.ClearRows()
```

Removes all rows and clears selection.

```
form.table1.ClearRows()
? form.table1.RowCount
```

Result Prints 0.

WID-025 / TableView.ClearColumns**GUI CONTROL: TABLEVIEW · SET-CONTROLS**`form.table1.ClearColumns()`

Removes columns and rows.

```
form.table1.ClearColumns()
? form.table1.ColumnCount
```

Result Prints 0.**WID-026 / TableView.SetCell****GUI CONTROL: TABLEVIEW · SET-CONTROLS**`form.table1.SetCell(rowIndex, columnIndexOrName, value)`

Changes one cell using zero-based row indexing.

```
? form.table1.SetCell(0, "name", "Kokum")
```

Result Prints .T. and changes the cell.**WID-027 / TableView.GetCell****GUI CONTROL: TABLEVIEW · SET-CONTROLS**`form.table1.GetCell(rowIndex, columnIndexOrName)`

Returns one cell or NULL.

```
? form.table1.GetCell(0, "name")
```

Result Prints the first row's name.**WID-028 / PropertyGrid.AddProperty****GUI CONTROL: PROPERTYGRID · SET-CONTROLS**`form.grid1.AddProperty(category, name, caption, value [, type [, readOnly]])`

Adds one uniquely named property.

```
? form.grid1.AddProperty("General", "name", "Name", "Anil", "string", .F.)
```

Result Prints .T. and displays Name = Anil.**WID-029 / PropertyGrid.SetValue****GUI CONTROL: PROPERTYGRID · SET-CONTROLS**`form.grid1.SetValue(name, value)`

Changes an existing property value.

```
? form.grid1.SetValue("name", "Kokum")
```

Result Prints .T. and displays Kokum.**WID-030 / PropertyGrid.GetValue****GUI CONTROL: PROPERTYGRID · SET-CONTROLS**`form.grid1.GetValue(name)`

Returns a property value or NULL.

```
? form.grid1.GetValue("name")
```

Result Prints the current name value.**WID-031 / PropertyGrid.RemoveProperty****GUI CONTROL: PROPERTYGRID · SET-CONTROLS**`form.grid1.RemoveProperty(name)`

Removes a property by name.

```
? form.grid1.RemoveProperty("name")
```

Result Prints .T. when name existed.

WID-032 / PropertyGrid.Clear**GUI CONTROL: PROPERTYGRID · SET-CONTROLS**`form.grid1.Clear()`

Removes all categories and properties.

```
form.grid1.Clear()
? form.grid1.PropertyCount
```

Result Prints 0.**Timer and GUI-safe asynchronous work**

Connect a Button click and Timer.timerTick slot. Start a retained TASK in the click slot and return promptly. In timerTick, test IsCompleted; read Result only when complete and update widgets there. Do not call blocking AWAIT in a responsive UI event or mutate live widgets from a worker. See the checked template `examples/kokum_029_url_api/gui_async_poll.kokum` in the baseline for the complete retained-task lifecycle.

```
&& Inside the Timer slot; pending was created by a click slot.
IF pending != NULL
  IF pending.IsCompleted
    TRY
      form.statusBar1.Text = STR(pending.Result())
    CATCH error
      form.statusBar1.Text = "Request failed"
    ENDTRY
    pending = NULL
  ENDIF
ENDIF
```

NOTE This is a contextual slot pattern, not a complete form: define PUBLIC pending AS TASK outside generated blocks, supply an async function and retain the task until completion. Closing a form does not automatically cancel its task.

10 / REFERENCE

Database commands and data components

Distinguish built-in Remote FlatDB calls from configured DATABASE receiver methods.

SOURCE BASIS [S10] Remote FlatDB, provider contracts and JSON-to-table implementation.

Remote FlatDB local lab

Build the optional Linux kokum-flatdb-server target if necessary. Work in a new disposable directory. Generate one key, retain its exact filename and bytes, and start the listener. Separate clients need a securely copied matching key; the key is not a password to type in source. Use --create only for initial setup, then omit it on ordinary restarts.

```
make kokum-flatdb-server
# In a separate lab directory with the binary on PATH:
openssl rand -out demo.kkey 32
chmod 600 demo.kkey
kokum-flatdb-server --database demo.kfdb --key demo.kkey \
  --listen 127.0.0.1 --port 9437 --create
```

Keep the server running and place this kdb.conf next to the source/project or deployed app, with demo.kkey relative to the configuration file.

```
[connection]
host = 127.0.0.1
port = 9437
database = demo.kfdb
key_file = demo.kkey
connect_timeout_ms = 5000
request_timeout_ms = 30000
max_message_bytes = 16777216
```

Kokum fragment — inside Main

```
KCONNECT USING kdb.conf
IF KCONNECTED()
  ? "connected"
  KDISCONNECT()
ELSE
  ? "Remote database is unavailable"
ENDIF
```

Unauthorized remote denial is intentionally generic. KEXECUTE and KQUERY take one SQL STRING each in this baseline; they do not accept a parameter ARRAY. Use the DATABASE provider methods below when that API is appropriate. Do not concatenate untrusted values into ad-hoc FlatDB SQL.

BI-067 / KCONNECT

EXTERNAL FLATDB SERVICE

KCONNECT(path) or KCONNECT USING path

Connects the Remote FlatDB client using a configuration-file path.

```
KCONNECT USING kdb.conf
? KCONNECTED()
```

Result Prints .T. when the configured server accepts the connection.

NOTE Requires the Remote FlatDB setup. KEXECUTE and KQUERY take exactly one SQL string; their signatures have no parameter-array argument.

BI-068 / KCONNECTED**EXTERNAL FLATDB SERVICE**

KCONNECTED()

Reports whether the process-level Remote FlatDB client is connected.

```
KCONNECT USING kdb.conf
? KCONNECTED()
KDISCONNECT()
```

Result Prints .T. after a successful authenticated connection.

NOTE Requires the Remote FlatDB setup. KEXECUTE and KQUERY take exactly one SQL string; their signatures have no parameter-array argument.

BI-069 / KEXECUTE**EXTERNAL FLATDB SERVICE**

KEXECUTE(sql)

Executes a non-query SQL statement through the Remote FlatDB connection.

```
KCONNECT USING kdb.conf
IF KCONNECTED()
    KEXECUTE("CREATE TABLE IF NOT EXISTS items (id INTEGER)")
    ? KEXECUTE("INSERT INTO items(id) VALUES (1)")
    KDISCONNECT()
ENDIF
```

Result Prints 1 affected row after connecting.

NOTE Requires the Remote FlatDB setup. KEXECUTE and KQUERY take exactly one SQL string; their signatures have no parameter-array argument.

BI-070 / KQUERY**EXTERNAL FLATDB SERVICE**

KQUERY(sql)

Executes a query and returns an ARRAY of row MAPs.

```
KCONNECT USING kdb.conf
IF KCONNECTED()
    KEXECUTE("CREATE TABLE IF NOT EXISTS items (id INTEGER)")
    KEXECUTE("DELETE FROM items")
    KEXECUTE("INSERT INTO items(id) VALUES (7)")
    ? JSONENCODE(KQUERY("SELECT id FROM items"))
    KDISCONNECT()
ENDIF
```

Result Prints [{"id":7}]. Uses a disposable test database.

NOTE Requires the Remote FlatDB setup. KEXECUTE and KQUERY take exactly one SQL string; their signatures have no parameter-array argument.

BI-071 / KDISCONNECT**EXTERNAL FLATDB SERVICE**

KDISCONNECT()

Closes the Remote FlatDB connection.

```
KCONNECT USING kdb.conf
IF KCONNECTED()
    ? KDISCONNECT()
    ? KCONNECTED()
ENDIF
```

Result Prints .T. then .F. after a successful initial connection.

NOTE Requires the Remote FlatDB setup. KEXECUTE and KQUERY take exactly one SQL string; their signatures have no parameter-array argument.

SQLite component and sample tables

In a Web GUI project, add a SQLite component named dbLocal. Set databaseFile to \${APP_DIR}/data/reference.sqlite3, openMode to readwrite-create, autoOpen to true and createParentDirectory to true. The designer creates PUBLIC dbLocal AS DATABASE. Add the following helper outside your slots and call PrepareDatabase() before a database example. These are disposable tables.

```
PROCEDURE PrepareDatabase()
  dbLocal.Open()
  dbLocal.Execute( ;
    "CREATE TABLE IF NOT EXISTS people " + ;
    "(id INTEGER PRIMARY KEY, name TEXT)" )
  dbLocal.Execute("DELETE FROM people")
  dbLocal.Execute( ;
    "INSERT INTO people(id,name) VALUES (?,?)", [1,"Anil"])
  dbLocal.Execute( ;
    "CREATE TABLE IF NOT EXISTS log (message TEXT)" )
ENDPROC
```

Paste a DATABASE method fragment inside a bound slot, after PrepareDatabase(). Do not declare dbLocal as an uninitialized local variable. For TigerDB-specific entries, add a configured TigerDB component dbTiger with access to an existing sales database. PostgreSQL uses the same common SQL methods through its own configured component; provider-only properties may be NULL or inapplicable on SQLite.

LANG-070 / CREATE TABLE ... ON DATA ... FROM

GUI/DATABASE PROJECT · SET-DATABASE

```
CREATE TABLE name ON DATA databaseExpr FROM jsonExpr
```

Creates a provider-neutral table from a MAP, ARRAY of MAPs, or JSON string and inserts records.

```
LOCAL rows AS ARRAY
rows = [{"name": "Anil", "city": "Ratnagiri"}]
dbLocal.Execute("DROP TABLE IF EXISTS people_import")
CREATE TABLE people_import ON DATA dbLocal FROM rows
```

Result Creates people_import with rec_no and inferred columns, then inserts one row.

NOTE Uses the disposable SQLite database from SET-DATABASE. DROP removes only this example table before a repeat run.

LANG-071 / SCHEMA ONLY

GUI/DATABASE PROJECT · SET-DATABASE

```
CREATE TABLE ... FROM source SCHEMA ONLY
```

Creates the inferred table structure without inserting the source rows.

```
dbLocal.Execute("DROP TABLE IF EXISTS people_template")
CREATE TABLE people_template ;
  ON DATA dbLocal ;
  FROM [{"name": "sample", "age": 1}] ;
  SCHEMA ONLY
```

Result Creates people_template with no inserted source records.

NOTE Creates a fresh sample table without inserting the source row.

API-001 / DATABASE.Open

SQLITE GUI SETUP · SET-DATABASE

```
Open()
```

Open the configured database connection.

```
? dbLocal.Open()
```

Result Prints .T. and opens the configured connection.

API-002 / DATABASE.Close**SQLITE GUI SETUP · SET-DATABASE**

Close()

Close the database connection.

```
dbLocal.Close()
? dbLocal.IsOpen
```

Result Prints .F.; repeated Close() is safe.**API-003 / DATABASE.Begin****SQLITE GUI SETUP · SET-DATABASE**

Begin()

Begin a transaction.

```
dbLocal.Begin()
? "transaction started"
dbLocal.Rollback()
```

Result Starts then rolls back an otherwise empty transaction.**API-004 / DATABASE.Commit****SQLITE GUI SETUP · SET-DATABASE**

Commit()

Commit the active transaction.

```
dbLocal.Begin()
dbLocal.Execute("INSERT INTO log(message) VALUES (?)", ["ok"])
dbLocal.Commit()
```

Result Commits the inserted row.**API-005 / DATABASE.Rollback****SQLITE GUI SETUP · SET-DATABASE**

Rollback()

Roll back the active transaction.

```
dbLocal.Begin()
dbLocal.Execute("INSERT INTO log(message) VALUES (?)", ["cancelled"])
dbLocal.Rollback()
```

Result The inserted row is not committed.**API-006 / DATABASE.Ping****EXTERNAL TIGERDB SETUP · SET-DATABASE**

Ping()

Ping a TigerDB or PostgreSQL server.

```
? JSONENCODE(dbTiger.Ping())
```

Result Prints the TigerDB/PostgreSQL ping response.**NOTE** Requires a configured TigerDB or PostgreSQL server; do not use the SQLite dbLocal setup for this call.**API-007 / DATABASE.UseDatabase****EXTERNAL TIGERDB SETUP · SET-DATABASE**

UseDatabase(name AS STRING)

Select a TigerDB database.

```
? JSONENCODE(dbTiger.UseDatabase("sales"))
```

Result Selects the TigerDB sales database.**NOTE** TigerDB-specific; use an existing permitted database name.

API-008 / DATABASE.Raw

EXTERNAL TIGERDB SETUP · SET-DATABASE

Raw(request AS STRING)

Send a raw TigerDB protocol request.

```
? JSONENCODE(dbTiger.Raw(JSONENCODE({"action": "ping"})))
```

Result Prints the raw TigerDB protocol response.

NOTE TigerDB-specific. Raw takes a JSON STRING, so encode a MAP first. It is not a provider-neutral arbitrary-call API.

API-009 / DATABASE.CreateTableFromJson

SQLITE GUI SETUP · SET-DATABASE

CreateTableFromJson(tableName AS STRING, source AS JSON [, schemaOnly AS LOGICAL])

Create a table from JSON and return the inserted row count.

```
dbLocal.Execute("DROP TABLE IF EXISTS imported_people")
? dbLocal.CreateTableFromJson(
    "imported_people", [{"name": "Anil", "age": 55}], .F.)
```

Result Creates imported_people and prints 1. Use a fresh table name.

NOTE This example deliberately recreates a sample table; use only the disposable lab database.

API-010 / DATABASE.Execute

SQLITE GUI SETUP · SET-DATABASE

Execute(sql AS STRING [, parameters AS ARRAY])

Execute parameterized SQL.

```
? dbLocal.Execute("INSERT INTO log(message) VALUES (?)", ["Anil"])
```

Result Prints 1. User data is supplied through a parameter array.

API-011 / DATABASE.Query

SQLITE GUI SETUP · SET-DATABASE

Query(sql AS STRING [, parameters AS ARRAY])

Return all result rows as maps.

```
LOCAL rows AS ARRAY
rows = dbLocal.Query("SELECT * FROM people ORDER BY id")
? TYPEOF(rows), LEN(rows)
```

Result Prints ARRAY and the row count.

API-012 / DATABASE.QueryOne

SQLITE GUI SETUP · SET-DATABASE

QueryOne(sql AS STRING [, parameters AS ARRAY])

Return the first row or NULL.

```
LOCAL row
row = dbLocal.QueryOne("SELECT name FROM people WHERE id = ?", [1])
IF row != NULL
    ? row["name"]
ENDIF
```

Result Prints Anil when id 1 exists. Checks for NULL before reading a field.

API-013 / DATABASE.QueryValue

SQLITE GUI SETUP · SET-DATABASE

QueryValue(sql AS STRING [, parameters AS ARRAY])

Return the first column of the first row.

```
? dbLocal.QueryValue("SELECT COUNT(*) FROM people")
```

Result Prints the scalar count.

API-014 / DATABASE.QueryTable

SQLITE GUI SETUP · SET-DATABASE

`QueryTable(sql AS STRING [, parameters AS ARRAY [, headers AS LOGICAL]])`

Return a two-dimensional TableView-compatible array.

```
LOCAL tableData AS ARRAY
tableData = dbLocal.QueryTable( ;
    "SELECT id, name FROM people", NULL, .T.)
? JSONENCODE(tableData)
```

Result Returns a two-dimensional array including its header row.

NOTE Only configured local SQLite examples were executed for this book. Ping, UseDatabase and Raw entries require an external server and operator-supplied connection settings. DATABASE.Raw takes a JSON STRING, not a MAP.

11 / REFERENCE

Graph and image APIs

Native resource handles expose methods that the UI can render without moving resource ownership into JavaScript.

SOURCE BASIS [S11] GRAPH and IMAGE implementation contracts.

Graph setup

Add a Graph control named graphSales. Before each independent graph example, restart the form or run this preparation inside a slot. The native GRAPH handle is the generated graphSales value, not form.graphSales for these methods.

```
graphSales.Clear()
graphSales.SetType("line")
graphSales.SetData(["Jan", "Feb", "Mar"], [10, 20, 30], "Sales")
graphSales.AddSeries("Target", [12, 18, 25])
```

API-015 / GRAPH.Clear

GRAPH GUI SETUP · SET-GRAPH

Clear()

Remove all graph data.

```
graphSales.Clear()
```

Result All labels, series, and candle data are removed.

API-016 / GRAPH.ClearData

GRAPH GUI SETUP · SET-GRAPH

ClearData()

Remove labels, series, and candle data.

```
graphSales.ClearData()
```

Result Same data-clearing result as Clear().

API-017 / GRAPH.Refresh

GRAPH GUI SETUP · SET-GRAPH

Refresh()

Force a browser redraw.

```
graphSales.Refresh()
```

Result The browser redraws the graph.

API-018 / GRAPH.SetType

GRAPH GUI SETUP · SET-GRAPH

SetType(type AS STRING)

Set line, bar, pie, or candle mode.

```
graphSales.SetType("line")
```

Result Graph mode becomes line.

API-019 / GRAPH.SetGraphType

GRAPH GUI SETUP · SET-GRAPH

SetGraphType(type AS STRING)

Set line, bar, pie, or candle mode.

```
graphSales.SetGraphType("bar")
```

Result Graph mode becomes bar.

API-020 / GRAPH.SetTitle**GRAPH GUI SETUP · SET-GRAPH**

```
SetTitle(title AS STRING)
```

Set the graph title.

```
graphSales.SetTitle("Quarterly Sales")
```

Result The title is displayed above the graph.

API-021 / GRAPH.SetData**GRAPH GUI SETUP · SET-GRAPH**

```
SetData(labels AS ARRAY, values AS ARRAY [, seriesName AS STRING [, color AS STRING]])
```

Replace labels and one complete data series; candle graphs accept SetData(candles).

```
graphSales.SetData(["Q1","Q2"], [120,150], "Sales")
```

Result Replaces labels and one complete series.

API-022 / GRAPH.SupplyData**GRAPH GUI SETUP · SET-GRAPH**

```
SupplyData(labels AS ARRAY, values AS ARRAY [, seriesName AS STRING [, color AS STRING]])
```

Alias for SetData.

```
graphSales.SupplyData(["Q1","Q2"], [120,150], "Sales")
```

Result Alias of SetData; the series is loaded.

API-023 / GRAPH.SetLabels**GRAPH GUI SETUP · SET-GRAPH**

```
SetLabels(labels AS ARRAY)
```

Replace graph labels.

```
graphSales.SetLabels(["Jan","Feb","Mar"])
```

Result Graph labels are replaced.

API-024 / GRAPH.SetSeries**GRAPH GUI SETUP · SET-GRAPH**

```
SetSeries(series AS ARRAY)
```

Replace all graph series.

```
graphSales.SetSeries([{"name":"Sales","values":[10,20,30]}])
```

Result All graph series are replaced.

API-025 / GRAPH.AddSeries**GRAPH GUI SETUP · SET-GRAPH**

```
AddSeries(name AS STRING, values AS ARRAY [, color AS STRING])
```

Add a named series.

```
graphSales.AddSeries("Target", [12,18,25])
```

Result Loads the Target series values.

API-026 / GRAPH.RemoveSeries**GRAPH GUI SETUP · SET-GRAPH**

```
RemoveSeries(name AS STRING)
```

Remove a named series.

```
? graphSales.RemoveSeries("Target")
```

Result Prints .T. when Target existed and was removed.

API-027 / GRAPH.AddPoint**GRAPH GUI SETUP · SET-GRAPH**

AddPoint(series AS STRING, label AS ANY, value AS NUMBER)

Append one point to a series.

```
graphSales.AddPoint("Sales", "Apr", 35)
```

Result Appends label Apr and value 35.

API-028 / GRAPH.SetPie**GRAPH GUI SETUP · SET-GRAPH**

SetPie(labels AS ARRAY, values AS ARRAY)

Replace pie-chart data.

```
graphSales.SetType("pie")
graphSales.SetPie(["A", "B"], [60, 40])
```

Result Displays two pie slices.

API-029 / GRAPH.AddSlice**GRAPH GUI SETUP · SET-GRAPH**

AddSlice(label AS ANY, value AS NUMBER)

Append one pie slice.

```
graphSales.SetType("pie")
graphSales.SetPie(["A", "B"], [60, 40])
graphSales.AddSlice("Cash", 10)
```

Result Appends one pie slice.

API-030 / GRAPH.SetCandles**GRAPH GUI SETUP · SET-GRAPH**

SetCandles(candles AS ARRAY)

Replace candlestick data.

```
graphSales.SetType("candle")
graphSales.SetCandles(["01 Sep", 102, 111, 98, 108])
```

Result Loads one OHLC candle.

API-031 / GRAPH.AddCandle**GRAPH GUI SETUP · SET-GRAPH**

AddCandle(label AS ANY, open AS NUMBER, high AS NUMBER, low AS NUMBER, close AS NUMBER)

Append one candlestick record.

```
graphSales.SetType("candle")
graphSales.AddCandle("02 Sep", 108, 115, 103, 105)
```

Result Appends one OHLC candle.

API-032 / GRAPH.SetOption**GRAPH GUI SETUP · SET-GRAPH**

SetOption(name AS STRING, value AS ANY)

Change one graph option.

```
graphSales.SetOption("showLegend", .T.)
```

Result The legend option becomes .T.

API-033 / GRAPH.SetLegend**GRAPH GUI SETUP · SET-GRAPH**`SetLegend(visible AS LOGICAL)`

Show or hide the graph legend.

```
graphSales.SetLegend(.T.)
```

Result Shows the legend.

API-034 / GRAPH.ShowLegend**GRAPH GUI SETUP · SET-GRAPH**`ShowLegend(visible AS LOGICAL)`

Show or hide the graph legend.

```
graphSales.ShowLegend(.F.)
```

Result Hides the legend.

API-035 / GRAPH.SetGrid**GRAPH GUI SETUP · SET-GRAPH**`SetGrid(visible AS LOGICAL)`

Show or hide Cartesian grid lines.

```
graphSales.SetGrid(.T.)
```

Result Shows Cartesian grid lines.

API-036 / GRAPH.ShowGrid**GRAPH GUI SETUP · SET-GRAPH**`ShowGrid(visible AS LOGICAL)`

Show or hide Cartesian grid lines.

```
graphSales.ShowGrid(.F.)
```

Result Hides Cartesian grid lines.

API-037 / GRAPH.SetAnimation**GRAPH GUI SETUP · SET-GRAPH**`SetAnimation(enabled AS LOGICAL)`

Enable or disable redraw animation.

```
graphSales.SetAnimation(.T.)
```

Result Enables redraw animation.

API-038 / GRAPH.SetSmooth**GRAPH GUI SETUP · SET-GRAPH**`SetSmooth(enabled AS LOGICAL)`

Enable or disable smoothed line segments.

```
graphSales.SetSmooth(.T.)
```

Result Uses smoothed line segments.

API-039 / GRAPH.SetStacked**GRAPH GUI SETUP · SET-GRAPH**`SetStacked(enabled AS LOGICAL)`

Enable or disable stacked bars.

```
graphSales.SetStacked(.T.)
```

Result Enables stacking for bar-mode rendering.

API-040 / GRAPH.SetLineWidth**GRAPH GUI SETUP · SET-GRAPH**

SetLineWidth(width AS NUMBER)

Set line and candle stroke width.

```
graphSales.SetLineWidth(2.5)
```

Result Sets stroke width to 2.5.

API-041 / GRAPH.SetMaxPoints**GRAPH GUI SETUP · SET-GRAPH**

SetMaxPoints(count AS INTEGER)

Set the maximum retained point count.

```
graphSales.SetMaxPoints(100)
```

Result Retains at most 100 points.

API-042 / GRAPH.SetMaximumPoints**GRAPH GUI SETUP · SET-GRAPH**

SetMaximumPoints(count AS INTEGER)

Set the maximum retained point count.

```
graphSales.SetMaximumPoints(100)
```

Result Alias of SetMaxPoints.

API-043 / GRAPH.SetXAxisTitle**GRAPH GUI SETUP · SET-GRAPH**

SetXAxisTitle(title AS STRING)

Set the horizontal-axis title.

```
graphSales.SetXAxisTitle("Quarter")
```

Result Displays the horizontal-axis title.

API-044 / GRAPH.SetYAxisTitle**GRAPH GUI SETUP · SET-GRAPH**

SetYAxisTitle(title AS STRING)

Set the vertical-axis title.

```
graphSales.SetYAxisTitle("Revenue")
```

Result Displays the vertical-axis title.

API-045 / GRAPH.SetPalette**GRAPH GUI SETUP · SET-GRAPH**

SetPalette(palette AS STRING)

Select theme, classic, cool, warm, market, or monochrome.

```
graphSales.SetPalette("cool")
```

Result Uses the cool palette.

API-046 / GRAPH.SetEmptyText**GRAPH GUI SETUP · SET-GRAPH**

SetEmptyText(text AS STRING)

Set the message shown when no data exists.

```
graphSales.SetEmptyText("No data available")
```

Result Shows this message when no data exists.

API-047 / GRAPH.Snapshot

GRAPH GUI SETUP · SET-GRAPH

Snapshot()

Return a copy of the current graph state.

```

LOCAL state AS MAP
state = graphSales.Snapshot()
? state["type"], state["title"]

```

Result Prints the current graph type and title.

Image setup

Add an Image control named imgLogo. Copy a valid PNG to images/kokum.png in the project; add it as a bundle resource targeting images/kokum.png. The cover mascot is an example asset, not something automatically extracted into your application. Use your own copy of the image. Native SetImage reads the configured application-relative resource.

```

[[resource]]
source = "images/kokum.png"
target = "images/kokum.png"

```

API-048 / IMAGE.SetImage

GUI: IMGLOGO · SET-IMAGE

SetImage(path AS STRING, keepAspectRatio AS LOGICAL)

Load a JPG, JPEG, or PNG image.

```

? imgLogo.SetImage("images/kokum.png", .T.)

```

Result Prints .T. and displays the PNG while preserving aspect ratio.

API-049 / IMAGE.Clear

GUI: IMGLOGO · SET-IMAGE

Clear()

Remove the current dynamic image.

```

imgLogo.Clear()

```

Result Removes the current dynamic image; a configured static source may be restored.

API-050 / IMAGE.Snapshot

GUI: IMGLOGO · SET-IMAGE

Snapshot()

Return the current Image state.

```

? JSONENCODE(imgLogo.Snapshot())

```

Result Prints a portable Image state map.

12 / REFERENCE

WebSocket APIs

Configure before connecting, handle the receive queue deliberately, and keep GUI callbacks short.

SOURCE BASIS [S12] Native WebSocket component, message queue and form integration.

Local WebSocket setup

Add a WebSocketClient component named wsFeed. Set url to ws://127.0.0.1:18082/echo, autoConnect=false and reconnect=false for this lab. A configured component supplies the generated native WEBSOCKET handle. Use the small Python echo server below in a separate terminal. It accepts only local demonstration traffic, unfragmented frames up to 64 KiB, and is not a production server.

Save both consecutive listings as one echo_server.py file.

```
import base64, hashlib, socketserver, struct

GUID = "258EAF55-E914-47DA-95CA-C5AB0DC85B11"

def readn(sock, size):
    data = b""
    while len(data) < size:
        part = sock.recv(size - len(data))
        if not part:
            raise EOFError()
        data += part
    return data

def send(sock, opcode, payload):
    n = len(payload)
    head = bytes([0x80 | opcode])
    head += bytes([n]) if n < 126 else b"\x7e" + struct.pack("!H", n)
    sock.sendall(head + payload)

class Echo(socketserver.BaseRequestHandler):
    def handle(self):
        s = self.request
        s.settimeout(30)
        try:
            header = b""
            while not header.endswith(b"\r\n\r\n"):
                header += readn(s, 1)
                if len(header) > 16384:
                    return
            fields = {}
            for line in header.decode("ascii").split("\r\n")[1:]:
                if ":" in line:
                    k, v = line.split(":", 1)
                    fields[k.lower()] = v.strip()
            key = fields["sec-websocket-key"]
            digest = hashlib.sha1((key + GUID).encode()).digest()
            accept = base64.b64encode(digest).decode()
            reply = "HTTP/1.1 101 Switching Protocols\r\n"
            reply += "Upgrade: websocket\r\nConnection: Upgrade\r\n"
            reply += "Sec-WebSocket-Accept: " + accept + "\r\n"
            if fields.get("sec-websocket-protocol"):
                protocol = fields["sec-websocket-protocol"].split(",")[0]
                reply += "Sec-WebSocket-Protocol: " + protocol.strip() + "\r\n"
```

echo_server.py — continued

```

        s.sendall((reply + "\r\n").encode())
    while True:
        a, b = readn(s, 2)
        if not (a & 0x80 and b & 0x80) or a & 0x70:
            return
        op, n = a & 15, b & 127
        if n == 127:
            return
        if n == 126:
            n = struct.unpack("!H", readn(s, 2))[0]
        if op >= 8 and n > 125:
            return
        mask = readn(s, 4)
        data = readn(s, n)
        data = bytes(x ^ mask[i % 4] for i, x in enumerate(data))
        if op in (1, 2):
            send(s, op, data)
        elif op == 9:
            send(s, 10, data)
        elif op == 8:
            send(s, 8, data)
        return
    except (OSError, EOFError, ValueError, KeyError):
        return

class Server(socketserver.ThreadingTCPServer):
    allow_reuse_address = True
    daemon_threads = True

with Server(("127.0.0.1", 18082), Echo) as server:
    print("Local echo: ws://127.0.0.1:18082/echo", flush=True)
    server.serve_forever()

```

Run in the terminal

```
python3 echo_server.py
```

Before independent networking snippets, close any prior connection, set the loopback URL and subprotocol json, then Connect(). Initial Connect waits for the handshake; send only after it succeeds. For receive snippets, send {"message":"hello"} first and allow the echo to arrive. Restart the form between incompatible configuration tests. Never send real authentication values over this plaintext lab connection; production credentials require verified wss://.

```

wsFeed.SetUrl("ws://127.0.0.1:18082/echo")
wsFeed.SetProtocols("json")
wsFeed.SetReconnect(.F., 1000, 0)
wsFeed.Connect()
wsFeed.SendJSON({"message":"hello"})

```

API-051 / WEBSOCKET.SetUrl**GUI: WSFEED · SET-WEBSOCKET**

```
SetUrl(url AS STRING)
```

Change the WebSocket URL while disconnected.

```
wsFeed.SetUrl("ws://127.0.0.1:18082/echo")
```

Result Stores the URL while disconnected.

NOTE Loopback-only lab URL; use verified wss:// for untrusted networks.

API-052 / WEBSOCKET.SetProtocols**GUI: WSFEED · SET-WEBSOCKET**

```
SetProtocols(protocols AS STRING)
```

Set the requested comma-separated subprotocol list.

```
wsFeed.SetProtocols("json,market-v1")
```

Result Requests the two subprotocols.

API-053 / WEBSOCKET.SetSubprotocols**GUI:** WSFEED · SET-WEBSOCKET

SetSubprotocols(protocols AS STRING)

Alias for SetProtocols.

```
wsFeed.SetSubprotocols("market-v1")
```

Result Alias of SetProtocols.**API-054 / WEBSOCKET.SetHeader****GUI:** WSFEED · SET-WEBSOCKET

SetHeader(name AS STRING, value AS STRING)

Set an HTTP upgrade header.

```
wsFeed.SetHeader("X-Client-Code", "ABC123")
```

Result Adds the custom upgrade header.**API-055 / WEBSOCKET.RemoveHeader****GUI:** WSFEED · SET-WEBSOCKET

RemoveHeader(name AS STRING)

Remove an HTTP upgrade header.

```
? wsFeed.RemoveHeader("X-Client-Code")
```

Result Prints .T. when the header existed.**API-056 / WEBSOCKET.ClearHeaders****GUI:** WSFEED · SET-WEBSOCKET

ClearHeaders()

Remove all custom upgrade headers.

```
wsFeed.ClearHeaders()
```

Result Removes all custom upgrade headers.**API-057 / WEBSOCKET.SetBearerToken****GUI:** WSFEED · SET-WEBSOCKET

SetBearerToken(token AS STRING [, header AS STRING])

Set a Bearer token, optionally using a custom header.

```
wsFeed.SetBearerToken("access-token")
```

Result Sets Authorization: Bearer access-token.**API-058 / WEBSOCKET.SetFeedToken****GUI:** WSFEED · SET-WEBSOCKET

SetFeedToken(token AS STRING [, header AS STRING])

Set a feed token, optionally using a custom header.

```
wsFeed.SetFeedToken("feed-token", "X-Feed-Token")
```

Result Sets the feed token header.**API-059 / WEBSOCKET.SetApiKey****GUI:** WSFEED · SET-WEBSOCKET

SetApiKey(token AS STRING [, header AS STRING])

Set an API key, optionally using a custom header.

```
wsFeed.SetApiKey("api-key", "X-API-Key")
```

Result Sets the API key header.

API-060 / WEBSOCKET.SetBasicAuth**GUI:** WSFEED · SET-WEBSOCKET

SetBasicAuth(username AS STRING, password AS STRING)

Set HTTP Basic authentication.

```
wsFeed.SetBasicAuth("user", "secret")
```

Result Configures HTTP Basic authentication.**API-061 / WEBSOCKET.SetCookie****GUI:** WSFEED · SET-WEBSOCKET

SetCookie(cookie AS STRING)

Set the Cookie header.

```
wsFeed.SetCookie("session=abc123")
```

Result Sets the Cookie upgrade header.**API-062 / WEBSOCKET.SetQueryParameter****GUI:** WSFEED · SET-WEBSOCKET

SetQueryParameter(name AS STRING, value AS STRING)

Set a URL query parameter.

```
wsFeed.SetQueryParameter("client", "kokum")
```

Result Adds or replaces ?client=kokum.**API-063 / WEBSOCKET.AddQueryParameter****GUI:** WSFEED · SET-WEBSOCKET

AddQueryParameter(name AS STRING, value AS STRING)

Set or replace a URL query parameter.

```
wsFeed.AddQueryParameter("token", "abc")
```

Result Adds or replaces the token query parameter.**API-064 / WEBSOCKET.RemoveQueryParameter****GUI:** WSFEED · SET-WEBSOCKET

RemoveQueryParameter(name AS STRING)

Remove a URL query parameter.

```
? wsFeed.RemoveQueryParameter("token")
```

Result Prints .T. when the parameter existed.**API-065 / WEBSOCKET.ClearQueryParameters****GUI:** WSFEED · SET-WEBSOCKET

ClearQueryParameters()

Remove all configured query parameters.

```
wsFeed.ClearQueryParameters()
```

Result Removes all configured URL query parameters.**API-066 / WEBSOCKET.SetReconnect****GUI:** WSFEED · SET-WEBSOCKET

SetReconnect(enabled AS LOGICAL [, delayMs AS INTEGER [, maxAttempts AS INTEGER]])

Configure automatic reconnection.

```
wsFeed.SetReconnect(.T., 1000, 10)
```

Result Enables reconnect with 1-second delay and 10 attempts.

API-067 / WEBSOCKET.Connect**GUI: WSFEED · SET-WEBSOCKET**`Connect([url AS STRING])`

Connect to the configured WebSocket endpoint, optionally replacing its URL.

```
? wsFeed.Connect()
```

Result Prints .T. after the initial connection opens; a failed connection raises an error.**NOTE** Initial Connect can block during DNS, connection and handshake. Subsequent messaging is worker-backed; keep GUI responsiveness in mind.**API-068 / WEBSOCKET.Close****GUI: WSFEED · SET-WEBSOCKET**`Close([code AS INTEGER [, reason AS STRING [, timeoutMs AS INTEGER]])`

Close the WebSocket connection.

```
? wsFeed.Close(1000, "Application shutdown", 3000)
```

Result Performs a normal close and prints success.**NOTE** Connect first. Run receive examples after `SendJSON({"message":"hello"})` against the local echo lab.**API-069 / WEBSOCKET.Disconnect****GUI: WSFEED · SET-WEBSOCKET**`Disconnect([code AS INTEGER [, reason AS STRING [, timeoutMs AS INTEGER]])`

Alias for Close.

```
? wsFeed.Disconnect(1000, "Done", 3000)
```

Result Alias of Close.**NOTE** Connect first. Run receive examples after `SendJSON({"message":"hello"})` against the local echo lab.**API-070 / WEBSOCKET.SendText****GUI: WSFEED · SET-WEBSOCKET**`SendText(text AS STRING)`

Queue a UTF-8 text message.

```
? wsFeed.SendText("hello")
```

Result Queues a UTF-8 text frame and prints .T.**NOTE** Connect first. Run receive examples after `SendJSON({"message":"hello"})` against the local echo lab.**API-071 / WEBSOCKET.SendJSON****GUI: WSFEED · SET-WEBSOCKET**`SendJSON(value AS JSON)`

Encode and queue a JSON message.

```
? wsFeed.SendJSON({"action":"subscribe","symbols":["NIFTY"]})
```

Result Encodes and queues a JSON frame.**NOTE** Connect first. Run receive examples after `SendJSON({"message":"hello"})` against the local echo lab.**API-072 / WEBSOCKET.SendBinary****GUI: WSFEED · SET-WEBSOCKET**`SendBinary(data AS ARRAY)`

Queue an ARRAY of byte values 0..255.

```
? wsFeed.SendBinary([1,2,3,4])
```

Result Queues four binary bytes.**NOTE** Connect first. Run receive examples after `SendJSON({"message":"hello"})` against the local echo lab.

API-073 / WEBSOCKET.Ping**GUI: WSFEED · SET-WEBSOCKET**

Ping([payload AS STRING])

Send an optional WebSocket ping payload.

```
? wsFeed.Ping("health")
```

Result Queues a ping frame.**NOTE** Connect first. Run receive examples after SendJSON({"message":"hello"}) against the local echo lab.**API-074 / WEBSOCKET.Wait****GUI: WSFEED · SET-WEBSOCKET**

Wait([timeoutMs AS INTEGER])

Wait for connection or queued activity.

```
? wsFeed.Wait(500)
```

Result Waits up to 500 ms for connection or queued activity.**NOTE** Connect first. Run receive examples after SendJSON({"message":"hello"}) against the local echo lab.**API-075 / WEBSOCKET.Receive****GUI: WSFEED · SET-WEBSOCKET**

Receive([timeoutMs AS INTEGER])

Receive the next queued event or NULL.

```

LOCAL message
message = wsFeed.Receive(500)
IF message != NULL
    ? JSONENCODE(message)
ENDIF

```

Result Prints the next event map when available.**NOTE** Connect first. Run receive examples after SendJSON({"message":"hello"}) against the local echo lab.**API-076 / WEBSOCKET.Poll****GUI: WSFEED · SET-WEBSOCKET**

Poll([timeoutMs AS INTEGER])

Alias for Receive.

```

LOCAL message
message = wsFeed.Poll(0)

```

Result Alias of Receive; performs a nonblocking poll.**NOTE** Connect first. Run receive examples after SendJSON({"message":"hello"}) against the local echo lab.**API-077 / WEBSOCKET.ReceiveText****GUI: WSFEED · SET-WEBSOCKET**

ReceiveText([timeoutMs AS INTEGER])

Receive the next text message or NULL.

```

LOCAL text
text = wsFeed.ReceiveText(500)
IF text != NULL
    ? text
ENDIF

```

Result Prints the next text payload when available; a timeout produces no output.**NOTE** Connect first. Run receive examples after SendJSON({"message":"hello"}) against the local echo lab.

API-078 / WEBSOCKET.ReceiveJSON**GUI: WSFEED · SET-WEBSOCKET**

ReceiveJSON([timeoutMs AS INTEGER])

Receive and decode the next JSON message or return NULL.

```

LOCAL data AS JSON
data = wsFeed.ReceiveJSON(500)
? JSONENCODE(data)

```

Result Prints the next decoded JSON value or NULL.**NOTE** Connect first. Run receive examples after SendJSON({"message":"hello"}) against the local echo lab.**API-079 / WEBSOCKET.Drain****GUI: WSFEED · SET-WEBSOCKET**

Drain([limit AS INTEGER])

Return and clear queued messages, optionally up to a limit.

```

LOCAL batch AS ARRAY
batch = wsFeed.Drain(250)
? LEN(batch)

```

Result Returns and clears up to 250 queued events.**NOTE** Connect first. Run receive examples after SendJSON({"message":"hello"}) against the local echo lab.**API-080 / WEBSOCKET.ClearMessages****GUI: WSFEED · SET-WEBSOCKET**

ClearMessages()

Discard queued incoming messages.

```

wsFeed.ClearMessages()
? wsFeed.PendingMessages

```

Result Prints 0.**NOTE** Connect first. Run receive examples after SendJSON({"message":"hello"}) against the local echo lab.**API-081 / WEBSOCKET.Snapshot****GUI: WSFEED · SET-WEBSOCKET**

Snapshot()

Return connection counters and state.

```

? JSONENCODE(wsFeed.Snapshot())

```

Result Prints state/counter information without secrets.

13 / REFERENCE

HTTP GET, POST and microservices

Use URLGET and URLPOST for synchronous native HTTP/HTTPS requests; use existing tasks for worker-backed concurrency.

SOURCE BASIS [S13] HTTP client, redirect/security, async service and final parser contracts.

A runnable local Kokum service

Save the source and configuration below in the same lab directory. Start the server in a second terminal, then run the client examples. This loopback service has no production authentication, inbound TLS, persistence or request-forwarding policy. It deliberately echoes only demonstration input.

service.kokum

```
MODULE reference.service VERSION "1.0.0"
EXPORT FUNCTION Main
EXPORT FUNCTION Hello
EXPORT FUNCTION Echo
EXPORT FUNCTION Raw

FUNCTION Main AS INTEGER
  RETURN 0
ENDFUNC

FUNCTION Hello(request AS MAP) AS MAP
  RETURN {"Status": 200, ;
    "Json": {"message": "Hello from Kokum"}}
ENDFUNC

FUNCTION Echo(request AS MAP) AS MAP
  RETURN {"Status": 200, "Json": request["Json"]}
ENDFUNC

FUNCTION Raw(request AS MAP) AS MAP
  RETURN {"Status": 200, "Body": request["Body"], ;
    "Headers": {"Content-Type": "text/plain; charset=utf-8"}}
ENDFUNC
```

server.conf

```
[server]
listen = 127.0.0.1
port = 18080
workers = 2
queue_capacity = 32

[route hello]
method = GET
path = /hello
handler = Hello
parameters = 1

[route echo]
method = POST
path = /echo
handler = Echo
parameters = 1

[route raw]
method = POST
path = /raw
handler = Raw
parameters = 1
```

Run in the terminal

```
kokumc compile service.kokum -o service.kbc
kokumvm serve service.kbc --config server.conf
```

BI-105 / URLGET

LOCAL HTTP SERVICE SETUP

```
URLGET(url [, options])
```

Performs an HTTP or HTTPS GET and returns a response MAP.

```

LOCAL response AS MAP
response = URLGET("http://127.0.0.1:18080/hello")
? response["Status"], response["Json"]["message"]

```

Result Prints 200 and Hello from Kokum.

NOTE Start the complete Kokum test service from the HTTP chapter first. HTTP 4xx/5xx are response maps, not transport exceptions.

BI-106 / URLPOST

LOCAL HTTP SERVICE SETUP

```
URLPOST(url, body [, options])
```

Posts text, JSON, an encoded form or an empty body. MAP/ARRAY bodies default to JSON.

```

LOCAL response AS MAP
response = URLPOST( ;
    "http://127.0.0.1:18080/echo", {"name":"Anil"})
? response["Json"]["name"]

```

Result Prints Anil.

NOTE Start the complete Kokum test service from the HTTP chapter first. HTTP 4xx/5xx are response maps, not transport exceptions.

HTTP-001 / HTTP query and headers

LOCAL HTTP SERVICE SETUP

```
URLGET(url, options)
```

Use Query for encoding instead of joining untrusted strings into a URL.

```

LOCAL r AS MAP
r = URLGET("http://127.0.0.1:18080/hello", ;
    {"Query":{"name":"Anil Jagtap"}, ;
    "Headers":{"Accept":"application/json"}})
? r["Status"]

```

Result Prints 200.

HTTP-002 / HTTP text and form bodies

LOCAL HTTP SERVICE SETUP

```
URLPOST(url, body, {"BodyType": mode})
```

STRING defaults to text; a form body requires a MAP. The raw endpoint echoes the exact request body.

```

LOCAL r AS MAP
r = URLPOST("http://127.0.0.1:18080/raw", "hello")
? r["Body"]
r = URLPOST("http://127.0.0.1:18080/raw", ;
    {"name":"Anil","tag":["a","b"]}, {"BodyType":"form"})
? r["Body"]

```

Result Prints hello, then name=Anil&tag=a&tag=b.

HTTP-003 / HTTP response versus transport error

LOCAL HTTP SERVICE SETUP

```
TRY ... URLGET(...) ... CATCH error ... ENDTRY
```

A valid 404 returns a MAP. Failure to connect, verify TLS or parse a response is catchable.

```

LOCAL r AS MAP
TRY
    r = URLGET("http://127.0.0.1:18080/missing")
    ? r["Status"], r["Ok"]
CATCH error
    ? "Transport failed"
ENDTRY

```

Result Prints 404 .F. from the local service.

HTTP-004 / Async HTTP

COMPLETE KOKUM SOURCE

ASYNC FUNCTION ... RETURN URLGET(...)

Use a retained TASK for network work. Awaiting twice reads the cached result; it does not send a second request.

```

ASYNC FUNCTION Fetch(url AS STRING) AS MAP
  RETURN URLGET(url, {"TimeoutMs":5000})
ENDFUNC
FUNCTION Main AS INTEGER
  LOCAL pending AS TASK
  LOCAL r AS MAP
  pending = Fetch("http://127.0.0.1:18080/hello")
  r = AWAIT pending
  ? r["Status"]
  RETURN 0
ENDFUNC

```

Result Prints 200.

All 20 request options

Option	Type / default	Purpose
Query	MAP / none	Encoded query values; arrays repeat a key.
Headers	MAP / none	Caller headers, subject to validation and redirect stripping.
TimeoutMs	INTEGER / 30000	Whole admission/transport/redirect budget; excludes earlier task queue and body preparation.
ConnectTimeoutMs	INTEGER / 5000	Connection and TLS budget constrained by total deadline.
ReadTimeoutMs	INTEGER / 30000	Idle response-read budget.
MaxResponseBytes	INTEGER / 8388608	Bound retained response body.
FollowRedirects	LOGICAL / .T.	Follow supported redirects.
MaxRedirects	INTEGER / 5	Redirect limit; hard maximum 20.
VerifyTLS	LOGICAL / .T.	Verify certificate trust and server identity.
CAFile	STRING / system trust	Explicit PEM CA trust file.
BodyType	STRING / automatic	Omit to infer from the body; explicit text, json or form.
ContentType	STRING / inferred	Explicit media type; do not also specify Content-Type in Headers.
UserAgent	STRING / Kokum/0.29.0	User-Agent value.
TLSServerName	STRING / empty	Optional SNI and verified server-identity override.
BasicAuth	MAP / none	Username and Password helper; HTTPS by default.
BearerToken	STRING / none	Validated Bearer helper; HTTPS by default.
Cookie	STRING / none	Explicit request cookie; no cookie jar.
CookiePolicy	STRING / manual	manual or omit.
AllowCrossOriginBody	LOGICAL / .F.	Opt into replaying a body to a different origin.
AllowInsecureAuth	LOGICAL / .F.	Allow helper-generated credentials on plaintext HTTP; lab-only exception.

All 14 response fields

Field	Type	Meaning
Status	INTEGER	Numeric HTTP status, including non-2xx.
Reason	STRING	Reason phrase from the HTTP status line.
Ok	LOGICAL	True for a successful HTTP status.
Url	STRING	Final URL after redirects.
HttpVersion	STRING	HTTP/1.0 or HTTP/1.1 from the status line.
Headers	MAP	Convenient normalized header lookup.
HeaderValues	MAP	Per-name arrays preserving repeated header values.
ContentType	STRING	Parsed response media-type value.
Body	STRING	Returned response body.
ContentLength	INTEGER	Actual retained response-body byte count.
Json	JSON / NULL	Decoded supported JSON response, otherwise NULL.
JsonError	STRING	JSON decode diagnostic, when applicable.
RedirectCount	INTEGER	Number of followed redirects.
ElapsedMs	INTEGER	Elapsed admission/transport/redirect time.

Security and timeout rules

HTTPS verification is enabled by default. BasicAuth and BearerToken require HTTPS unless explicitly overridden; do not combine both helpers or mix a helper with a manual Authorization header. A cross-origin redirect permanently removes credentials and unknown/custom headers. HTTPS-to-HTTP redirects are always blocked. POST becomes GET on 301/302/303; 307/308 preserve the body, but cross-origin replay is blocked unless explicitly permitted.

CookiePolicy=manual sends only explicitly supplied cookies; omit suppresses them. There is no persistent cookie jar, automatic retry, cancellation API, connection pool, streaming or multipart upload. Synchronous system DNS cannot be preempted by the deadline. A valid HTTP 404/500 is still a response MAP; transport, TLS and parser faults are catchable errors. This client is not an SSRF destination sandbox.

14 / REFERENCE

Compiler, VM and IDE command line

The language runs through three public tools; the optional FlatDB server is a separate service binary.

SOURCE BASIS [S14] Current executable help and CLI implementation.

Prepare the toolchain lab

Use the two-module project from SET-MODULE as the working directory. Also save the first program from SET-MAIN as hello.kokum and main.kokum. Run the preparation below once. The commands assume correctly installed binaries on PATH; an uninstalled source build may require --assets and --kokumvm when starting the IDE.

```
kokumc compile hello.kokum -o main.kbc
kokumc compile-module math.kokum sample.math 1.0.0 -o math.kbc
kokumc compile-module app.kokum sample.app 1.0.0 -o app.kbc
kokumc bundle kokum.toml
kokumc project new --type web-gui \
  --name ReferenceDemo --directory reference-demo
kokumc build reference-demo/kokum.toml
kokumc form compile \
  reference-demo/ui/main_window.kform \
  reference-demo/src/slots.kokum -o Main.kres
kokumc compile reference-demo/src/slots.kokum -o slots.kbc
```

demo.kapp is the console module bundle. reference-demo/dist/referencedemo.kapp is the GUI bundle for web launches. Do not interchange them. The service command uses service.kbc from SET-HTTP. old-project refers to an actual legacy project that you supply; migration examples use --dry-run. Interactive/server commands remain running until closed or interrupted.

Compiler commands and options

CLI-001 / kokumc --version

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumc --version
```

Prints compiler version information.

```
kokumc --version
```

Result Prints Kokum 0.29.0 build identity.

CLI-002 / kokumc --about

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumc --about
```

Prints product/company information.

```
kokumc --about
```

Result Prints Kokum and TigerDB Solutions information.

CLI-003 / kokumc check

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumc check file.kokum
```

Runs lexical, parse, and semantic checks.

```
kokumc check hello.kokum
```

Result Exits 0 when the source is valid.

CLI-004 / kokumc compile**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc compile file.kokum -o file.kbc
```

Compiles source to portable bytecode.

```
kokumc compile hello.kokum -o main.kbc
```

Result Creates main.kbc.

CLI-005 / kokumc compile-module**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc compile-module file name version -o file.kbc
```

Compiles an explicitly named/versioned module.

```
kokumc compile-module math.kokum sample.math 1.0.0 -o math.kbc
```

Result Creates math.kbc with module identity.

CLI-006 / kokumc tokens**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc tokens file.kokum
```

Prints lexer tokens.

```
kokumc tokens main.kokum
```

Result Displays the token stream.

CLI-007 / kokumc ast**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc ast file.kokum
```

Prints the parsed abstract syntax tree.

```
kokumc ast main.kokum
```

Result Displays the AST.

CLI-008 / kokumc symbols**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc symbols file.kokum
```

Prints declarations and resolved symbols.

```
kokumc symbols main.kokum
```

Result Displays the symbol table.

CLI-009 / kokumc ir**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc ir file.kokum
```

Prints canonical IR.

```
kokumc ir main.kokum
```

Result Displays lowered IR.

CLI-010 / kokumc verify-ir**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc verify-ir file.kokum
```

Builds and verifies canonical IR.

```
kokumc verify-ir main.kokum
```

Result Exits 0 when IR invariants hold.

CLI-011 / kokumc run**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc run file.kokum
```

Runs source through the reference execution path.

```
kokumc run hello.kokum
```

Result Executes the program.

CLI-012 / kokumc run-ir**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc run-ir file.kokum
```

Runs the canonical-IR executor.

```
kokumc run-ir hello.kokum
```

Result Produces the same observable result.

CLI-013 / kokumc run-bytecode**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc run-bytecode file.kokum
```

Compiles and executes through bytecode/VM.

```
kokumc run-bytecode hello.kokum
```

Result Produces the same observable result.

CLI-014 / kokumc project new**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc project new --type TYPE --name NAME --directory PATH
```

Creates a project skeleton.

```
kokumc project new --type web-gui \  
  --name ReferenceDemo --directory reference-demo
```

Result Creates the Web GUI project in reference-demo.

NOTE Run once in a writable directory. Do not reuse a nonempty project directory.

CLI-015 / kokumc project info**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc project info [kokum.toml]
```

Prints project metadata.

```
kokumc project info reference-demo/kokum.toml
```

Result Displays project identity and targets.

CLI-016 / kokumc project check**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc project check [kokum.toml]
```

Validates project structure and sources.

```
kokumc project check reference-demo/kokum.toml
```

Result Exits 0 when valid.

CLI-017 / kokumc project graph**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc project graph [kokum.toml]
```

Prints the module/dependency graph.

```
kokumc project graph reference-demo/kokum.toml
```

Result Displays linked modules and dependencies.

CLI-018 / kokumc project migrate**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc project migrate [--dry-run] [--no-backup] project
```

Migrates a project manifest/layout.

```
kokumc project migrate --dry-run old-project
```

Result Reports changes without writing them.

CLI-019 / kokumc migrate-project**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc migrate-project ...
```

Compatibility alias for project migration.

```
kokumc migrate-project --dry-run old-project
```

Result Reports planned migration.

CLI-020 / kokumc build**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc build [kokum.toml]
```

Builds the configured application/project.

```
kokumc build reference-demo/kokum.toml
```

Result Creates project build outputs.

CLI-021 / kokumc build --standalone**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc build --standalone [kokum.toml] [-o executable]
```

Builds a self-contained executable launcher image.

```
kokumc build --standalone kokum.toml -o demo-app
```

Result Creates demo-app.

CLI-022 / kokumc restore**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc restore [kokum.toml]
```

Restores locked package dependencies.

```
kokumc restore kokum.toml
```

Result Populates the package cache/lock requirements.

CLI-023 / kokumc bundle**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc bundle [kokum.toml]
```

Builds an application bundle.

```
kokumc bundle kokum.toml
```

Result Creates the configured .kapp bundle.

CLI-024 / kokumc form check**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc form check form.kform
```

Validates a form document.

```
kokumc form check reference-demo/ui/main_window.kform
```

Result Exits 0 when the form is valid.

CLI-025 / kokumc form bind-check

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumc form bind-check form.kform slots.kokum
```

Checks event-slot bindings.

```
kokumc form bind-check \  
  reference-demo/ui/main_window.kform \  
  reference-demo/src/slots.kokum
```

Result Confirms every export resolves to a local routine.

CLI-026 / kokumc form compile

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumc form compile form.kform slots.kokum -o form.kres
```

Compiles form plus slots to a resource.

```
kokumc form compile \  
  reference-demo/ui/main_window.kform \  
  reference-demo/src/slots.kokum -o Main.kres
```

Result Creates Main.kres.

CLI-027 / kokumc form info

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumc form info form.kres
```

Prints compiled form information.

```
kokumc form info Main.kres
```

Result Displays form metadata.

CLI-028 / kokumc form verify

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumc form verify form.kres
```

Verifies compiled form integrity.

```
kokumc form verify Main.kres
```

Result Exits 0 when the resource is valid.

CLI-029 / kokumc form disassemble

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumc form disassemble form.kres
```

Prints compiled form contents.

```
kokumc form disassemble Main.kres
```

Result Displays the resource structure.

CLI-030 / kokumc form schema

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumc form schema [ControlType]
```

Lists all GUI controls or one control schema.

```
kokumc form schema TreeView
```

Result Displays TreeView properties and events.

CLI-031 / kokumc package create**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc package create module.kbc -o module.kpkg
```

Creates a package archive.

```
kokumc package create math.kbc -o math.kpkg
```

Result Creates math.kpkg.

CLI-032 / kokumc package info**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc package info module.kpkg
```

Prints package metadata.

```
kokumc package info math.kpkg
```

Result Displays package entries and identity.

CLI-033 / kokumc package verify**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc package verify module.kpkg
```

Verifies package structure and hashes.

```
kokumc package verify math.kpkg
```

Result Exits 0 when valid.

CLI-034 / kokumc package signing-id**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc package signing-id module.kpkg
```

Prints the package signing identity.

```
kokumc package signing-id math.kpkg
```

Result Displays the signing ID.

CLI-035 / kokumc package extract**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc package extract package index destination
```

Extracts one package member.

```
kokumc package extract math.kpkg 0 math-entry.kbc
```

Result Writes member 0 to the destination.

CLI-036 / kokumc bundle info**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc bundle info application.kapp
```

Prints bundle metadata.

```
kokumc bundle info demo.kapp
```

Result Displays bundle entries.

CLI-037 / kokumc bundle verify**SHELL; SETUP REQUIRED · SET-TOOLS**

```
kokumc bundle verify application.kapp
```

Verifies a bundle.

```
kokumc bundle verify demo.kapp
```

Result Exits 0 when valid.

CLI-038 / kokumc bundle extract**SHELL; SETUP REQUIRED · SET-TOOLS**`kokumc bundle extract application.kapp index destination`

Extracts one bundle entry.

`kokumc bundle extract demo.kapp 0 bundle-entry.bin`**Result** Writes entry 0.**CLI-039 / kokumc cache add****SHELL; SETUP REQUIRED · SET-TOOLS**`kokumc cache add module.kpkg [cache-directory]`

Adds a package to a cache.

`kokumc cache add math.kpkg reference-cache`**Result** Copies/registers the package.**CLI-040 / kokumc cache list****SHELL; SETUP REQUIRED · SET-TOOLS**`kokumc cache list [cache-directory]`

Lists cached packages.

`kokumc cache list reference-cache`**Result** Displays cached identities.**CLI-041 / kokumc language-server****SHELL; SETUP REQUIRED · SET-TOOLS**`kokumc language-server --stdio [--project PATH]`

Starts the editor language service over stdio.

`kokumc language-server --stdio --project reference-demo`**Result** Waits for language-server protocol input.**CLI-042 / kokumc ide****SHELL; SETUP REQUIRED · SET-TOOLS**`kokumc ide [IDE options]`

Starts the IDE through the compiler compatibility route.

`kokumc ide reference-demo`**Result** Opens the project in the browser IDE.**CLI-043 / kokumc edit****SHELL; SETUP REQUIRED · SET-TOOLS**`kokumc edit [editor options] [file.kokum]`

Starts the editor compatibility route.

`kokumc edit main.kokum`**Result** Opens main.kokum.**CLI-044 / kokumc web****SHELL; SETUP REQUIRED · SET-TOOLS**`kokumc web application.kapp [options]`

Runs a Web GUI bundle through the compiler compatibility route.

`kokumc web reference-demo/dist/referencedemo.kapp --no-browser`**Result** Starts the application host without opening a browser.

VM execution, inspection and linking

CLI-045 / kokumvm execute bytecode

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumvm file.kbc
```

Runs a bytecode module.

```
kokumvm main.kbc
```

Result Executes main.kbc.

CLI-046 / kokumvm execute bundle

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumvm application.kapp
```

Runs an application bundle.

```
kokumvm demo.kapp
```

Result Starts the application.

CLI-047 / kokumvm web

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumvm web application.kapp [options]
```

Runs a browser application host.

```
kokumvm web reference-demo/dist/referencedemo.kapp --no-browser
```

Result Starts the local Web host.

CLI-048 / kokumvm serve

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumvm serve application [--config server.conf]
```

Runs a configured application service host.

```
kokumvm serve service.kbc --config server.conf
```

Result Starts the configured service.

NOTE Use service.kbc and server.conf from SET-HTTP, not the console demo.kapp.

CLI-049 / kokumvm gui

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumvm gui [--backend ...] form.kres modules...
```

Runs a compiled native/headless form resource.

```
kokumvm gui --backend headless Main.kres slots.kbc
```

Result Runs the form in headless mode.

NOTE Compile the generated slots module first as shown in SET-TOOLS. This headless command verifies/dispatches GUI infrastructure; it is not the browser application launcher.

CLI-050 / kokumvm verify

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumvm verify file
```

Verifies bytecode, bundle, or standalone executable structure.

```
kokumvm verify demo.kapp
```

Result Exits 0 when valid.

CLI-051 / kokumvm info

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumvm info bundle-or-standalone
```

Prints bundle/standalone information.

```
kokumvm info demo.kapp
```

Result Displays embedded metadata.

CLI-052 / kokumvm disasm

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumvm disasm file.kbc
```

Disassembles bytecode.

```
kokumvm disasm main.kbc
```

Result Displays instructions and constants.

CLI-053 / kokumvm profile

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumvm profile file.kbc
```

Runs bytecode with profiler output.

```
kokumvm profile main.kbc
```

Result Displays execution profile data.

CLI-054 / kokumvm link

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumvm link entry-module module1.kbc ...
```

Links modules using the default entry.

```
kokumvm link sample.app app.kbc math.kbc
```

Result Links and runs/checks the module set.

CLI-055 / kokumvm link-entry

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumvm link-entry entry-module routine modules...
```

Links with an explicit entry routine.

```
kokumvm link-entry sample.app Main app.kbc math.kbc
```

Result Uses sample.app.Main as entry.

CLI-056 / kokumvm link-check

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokumvm link-check modules...
```

Checks whether modules form a resolvable set.

```
kokumvm link-check app.kbc math.kbc
```

Result Exits 0 when imports resolve.

IDE and supporting tool commands

CLI-057 / kokum-ide project

SHELL; SETUP REQUIRED · SET-TOOLS

`kokum-ide [project-directory | kokum.toml]`

Opens a project in the browser IDE.

```
kokum-ide reference-demo
```

Result Starts the loopback IDE and opens ReferenceDemo.

CLI-058 / kokum-ide --terminal

SHELL; SETUP REQUIRED · SET-TOOLS

`kokum-ide --terminal [editor-options] [file]`

Starts terminal editor mode.

```
kokum-ide --terminal main.kokum
```

Result Opens main.kokum in the terminal editor.

CLI-059 / kokum-ide server

SHELL; SETUP REQUIRED · SET-TOOLS

`kokum-ide server --config FILE`

Starts authenticated private-LAN IDE server mode.

```
kokum-ide server --config /etc/kokum/ide-lan.conf
```

Result Starts the configured LAN service.

NOTE The baseline LAN IDE authenticates per-user workspaces on a private IPv4 address. Build/Run are disabled in this LAN mode. Create users.csv and the configuration before starting; see the LAN setup.

CLI-060 / --no-browser

SHELL; SETUP REQUIRED · SET-TOOLS

`kokum-ide project --no-browser`

Suppresses automatic browser launch.

```
kokum-ide reference-demo --no-browser
```

Result Prints the per-launch authenticated URL without opening a browser.

CLI-061 / --port

SHELL; SETUP REQUIRED · SET-TOOLS

`kokum-ide project --port N`

Selects a loopback port; zero requests an automatic port.

```
kokum-ide reference-demo --port 0
```

Result Starts on an available port.

CLI-062 / --trace

SHELL; SETUP REQUIRED · SET-TOOLS

`kokum-ide project --trace`

Enables HTTP/WebSocket trace logging.

```
kokum-ide reference-demo --trace
```

Result Prints protocol activity.

CLI-063 / --headless-smoke**SHELL; SETUP REQUIRED · SET-TOOLS**`kokum-ide project --headless-smoke`

Validates backend/workspace without a browser.

```
kokum-ide reference-demo --headless-smoke \
  --assets /path/to/source/web/ide --kokumvm /path/to/kokumvm
```

Result Exits after the smoke validation.**NOTE** Use absolute paths to the current matching asset tree and VM when running an uninstalled build.**CLI-064 / kokumc --help****SHELL; SETUP REQUIRED · SET-TOOLS**`kokumc --help`

Show the compiler command families.

```
kokumc --help
```

Result Lists compiler command families and available options.**CLI-065 / kokumvm --help****SHELL; SETUP REQUIRED · SET-TOOLS**`kokumvm --help`

Show execution, linking and inspection commands.

```
kokumvm --help
```

Result Lists VM execution, verification and service commands.**CLI-066 / kokum-ide --help****SHELL; SETUP REQUIRED · SET-TOOLS**`kokum-ide --help`

Show IDE options.

```
kokum-ide --help
```

Result Lists IDE launch modes and configuration options.**CLI-067 / kokumvm --version****SHELL; SETUP REQUIRED · SET-TOOLS**`kokumvm --version`

Show the VM version and build identity.

```
kokumvm --version
```

Result Prints the current KokumVM version.**CLI-068 / kokumvm --about****SHELL; SETUP REQUIRED · SET-TOOLS**`kokumvm --about`

Show company information.

```
kokumvm --about
```

Result Prints KokumVM product and company information.**CLI-069 / kokum-ide --version****SHELL; SETUP REQUIRED · SET-TOOLS**`kokum-ide --version`

Show the IDE version.

```
kokum-ide --version
```

Result Prints the current IDE version.

CLI-070 / kokum-ide --about**SHELL; SETUP REQUIRED · SET-TOOLS**`kokum-ide --about`

Show company information.

```
kokum-ide --about
```

Result Prints IDE product and company information.

CLI-071 / kokum-ide --assets**SHELL; SETUP REQUIRED · SET-TOOLS**`kokum-ide reference-demo --assets /path/to/source/web/ide`

Choose the browser assets actually served by the IDE.

```
kokum-ide reference-demo --assets /path/to/source/web/ide
```

Result Starts the IDE using the explicitly selected frontend assets.

CLI-072 / kokum-ide --no-url**SHELL; SETUP REQUIRED · SET-TOOLS**`kokum-ide reference-demo --no-browser --no-url`

Suppress printing the per-launch secure URL.

```
kokum-ide reference-demo --no-browser --no-url
```

Result Starts the IDE without printing the launch URL.

CLI-073 / kokum-ide --timeout-ms**SHELL; SETUP REQUIRED · SET-TOOLS**`kokum-ide reference-demo --timeout-ms 60000`

Set the inactivity timeout in milliseconds.

```
kokum-ide reference-demo --timeout-ms 60000
```

Result Applies the configured IDE timeout; the IDE reports its launch settings.

CLI-074 / kokum-ide --recent**SHELL; SETUP REQUIRED · SET-TOOLS**`kokum-ide reference-demo --recent recent-projects.json`

Use a separate recent-project list.

```
kokum-ide reference-demo --recent recent-projects.json
```

Result Uses the specified recent-projects file.

CLI-075 / kokum-ide --kokumvm**SHELL; SETUP REQUIRED · SET-TOOLS**`kokum-ide reference-demo --kokumvm /path/to/kokumvm`

Choose the VM used by Run.

```
kokum-ide reference-demo --kokumvm /path/to/kokumvm
```

Result Uses the explicitly selected KokumVM executable for application launches.

CLI-076 / kokum-ide --no-restore**SHELL; SETUP REQUIRED · SET-TOOLS**`kokum-ide reference-demo --no-restore`

Do not restore per-project IDE state.

```
kokum-ide reference-demo --no-restore
```

Result Starts without restoring the previous IDE state.

CLI-077 / kokum-ide --no-create

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokum-ide reference-demo --no-create
```

Disable project creation in this IDE session.

```
kokum-ide reference-demo --no-create
```

Result Disables project creation for this IDE launch.

CLI-078 / kokum-flatdb-server

SHELL; SETUP REQUIRED · SET-TOOLS

```
kokum-flatdb-server --database demo.kfdb --key demo.kkey \
```

Start the separate Linux Remote FlatDB service; create the key first.

```
kokum-flatdb-server --database demo.kfdb --key demo.kkey \
  --listen 127.0.0.1 --port 9437 --create
```

Result Starts the loopback FlatDB listener; use --create only for initial setup.

Compatibility aliases and launch boundaries

The VM retains --verify, --disassemble, --profile, --bundle, --verify-bundle, --bundle-info, --link, --link-entry, --link-only and --gui-headless aliases. Compiler historical flags include --tokens, --ast, --check and --emit-bytecode. Prefer the canonical subcommands printed above for new scripts. Help output is the authority for platform- and command-specific options.

The local IDE binds to loopback and uses a per-launch token. LAN server mode binds to one explicit private IPv4 address, authenticates users from base-directory/users.csv, and isolates user workspaces. In the current LAN mode, Build/Run are disabled; do not promise the local-mode execution workflow over LAN. Do not expose the development server publicly. The supplied network manual and source LAN configuration guide describe the required users/configuration files.

15 / REFERENCE

Native property reference

Read properties with member syntax; methods require parentheses. The configured receiver must exist first.

SOURCE BASIS [S6, S10–S12] Current native property catalogue and resource implementations.

The following 90 readable properties match the source catalogue. A property's value depends on receiver state and provider. Not every database property has a meaningful value on every provider. These read examples do not grant permission to assign to read-only properties. GUI form schema properties and HTTP response MAP keys are separate interfaces.

DATABASE properties

Property / type	Meaning	Read example
P01-01 DATABASE.IsOpen LOGICAL	True while the database connection is open.	? dbLocal.IsOpen
P01-02 DATABASE.Provider STRING	Canonical provider name: sqlite, tigerdb, or postgresql.	? dbLocal.Provider
P01-03 DATABASE.Id STRING	Stable database component identifier.	? dbLocal.Id
P01-04 DATABASE.File STRING	SQLite database path when applicable.	? dbLocal.File
P01-05 DATABASE.Host STRING	Remote database host when applicable.	? dbLocal.Host
P01-06 DATABASE.Port INTEGER	Remote database port when applicable.	? dbLocal.Port
P01-07 DATABASE.Database STRING	Selected database name.	? dbLocal.Database
P01-08 DATABASE.Endpoint STRING	Resolved provider endpoint.	? dbLocal.Endpoint
P01-09 DATABASE.TLS LOGICAL	Whether TLS is enabled for the provider.	? dbLocal.TLS
P01-10 DATABASE.Schema STRING	Selected PostgreSQL schema.	? dbLocal.Schema
P01-11 DATABASE.SSLMode STRING	PostgreSQL SSL mode.	? dbLocal.SSLMode
P01-12 DATABASE.ApplicationName STRING	Provider application name.	? dbLocal.ApplicationName
P01-13 DATABASE.ReadOnly LOGICAL	Whether the component is configured read-only.	? dbLocal.ReadOnly
P01-14 DATABASE.ServerVersion INTEGER	Connected PostgreSQL server version number when available.	? dbLocal.ServerVersion
P01-15 DATABASE.BackendPid INTEGER	PostgreSQL backend process identifier when available.	? dbLocal.BackendPid
P01-16 DATABASE.RequestCount INTEGER	Number of provider requests issued.	? dbLocal.RequestCount
P01-17 DATABASE.LastElapsedMs	Duration of the latest provider request.	? dbLocal.LastElapsedMs

Property / type	Meaning	Read example
NUMBER		
P01-18 DATABASE.LastInsertId INTEGER	Latest generated row identifier when available.	? dbLocal.LastInsertId
P01-19 DATABASE.Changes INTEGER	Rows changed by the latest Execute operation.	? dbLocal.Changes

GRAPH properties

Property / type	Meaning	Read example
P02-01 GRAPH.Id STRING	Stable Graph control identifier.	? graphSales.Id
P02-02 GRAPH.Type STRING	line, bar, pie, or candle.	? graphSales.Type
P02-03 GRAPH.GraphType STRING	Alias for Type.	? graphSales.GraphType
P02-04 GRAPH.Title STRING	Displayed graph title.	? graphSales.Title
P02-05 GRAPH.ShowLegend LOGICAL	True when the legend is visible.	? graphSales.ShowLegend
P02-06 GRAPH.ShowGrid LOGICAL	True when Cartesian grid lines are visible.	? graphSales.ShowGrid
P02-07 GRAPH.XAxisTitle STRING	Horizontal-axis title.	? graphSales.XAxisTitle
P02-08 GRAPH.YAxisTitle STRING	Vertical-axis title.	? graphSales.YAxisTitle
P02-09 GRAPH.Palette STRING	Current graph colour palette.	? graphSales.Palette
P02-10 GRAPH.EmptyText STRING	Message shown when no graph data exists.	? graphSales.EmptyText
P02-11 GRAPH.Animation LOGICAL	Current redraw-animation setting.	? graphSales.Animation
P02-12 GRAPH.Smooth LOGICAL	Current line-smoothing setting.	? graphSales.Smooth
P02-13 GRAPH.Stacked LOGICAL	Current stacked-series setting.	? graphSales.Stacked
P02-14 GRAPH.LineWidth NUMBER	Current line and candle stroke width.	? graphSales.LineWidth
P02-15 GRAPH.MaximumPoints INTEGER	Maximum retained point count.	? graphSales.MaximumPoints
P02-16 GRAPH.MaxPoints INTEGER	Alias for MaximumPoints.	? graphSales.MaxPoints
P02-17 GRAPH.Labels ARRAY	Current graph labels.	? graphSales.Labels
P02-18 GRAPH.Series ARRAY	Current graph series.	? graphSales.Series

Property / type	Meaning	Read example
P02-19 GRAPH.Candles ARRAY	Current candlestick data.	? graphSales.Candles
P02-20 GRAPH.Options MAP	Current rendering options.	? graphSales.Options
P02-21 GRAPH.Revision INTEGER	Current graph-state revision.	? graphSales.Revision
P02-22 GRAPH.SeriesCount INTEGER	Number of graph series.	? graphSales.SeriesCount
P02-23 GRAPH.PointCount INTEGER	Number of graph points or candles.	? graphSales.PointCount

IMAGE properties

Property / type	Meaning	Read example
P03-01 IMAGE.Loaded LOGICAL	True after a dynamic image has been loaded.	? imgLogo.Loaded
P03-02 IMAGE.KeepAspectRatio LOGICAL	Current aspect-ratio policy.	? imgLogo.KeepAspectRatio
P03-03 IMAGE.ScaleMode STRING	fit or stretch.	? imgLogo.ScaleMode
P03-04 IMAGE.Source STRING	Current browser-facing image source.	? imgLogo.Source
P03-05 IMAGE.Path STRING	Current local source path.	? imgLogo.Path
P03-06 IMAGE.MimeType STRING	image/jpeg or image/png.	? imgLogo.MimeType
P03-07 IMAGE.Revision INTEGER	Current private-image revision.	? imgLogo.Revision
P03-08 IMAGE.Size INTEGER	Loaded image size in bytes.	? imgLogo.Size

WEBSOCKET properties

Property / type	Meaning	Read example
P04-01 WEBSOCKET.Id STRING	Stable WebSocket component identifier.	? wsFeed.Id
P04-02 WEBSOCKET.Url STRING	Current ws:// or wss:// URL.	? wsFeed.Url
P04-03 WEBSOCKET.State STRING	Current connection state.	? wsFeed.State
P04-04 WEBSOCKET.IsConnected LOGICAL	True while connected.	? wsFeed.IsConnected
P04-05 WEBSOCKET.Connected LOGICAL	Alias for IsConnected.	? wsFeed.Connected
P04-06 WEBSOCKET.IsRunning	True while the worker is active.	? wsFeed.IsRunning

Property / type	Meaning	Read example
LOGICAL		
P04-07 WEBSOCKET.PendingMessages INTEGER	Queued incoming message count.	? wsFeed.PendingMessages
P04-08 WEBSOCKET.PendingSends INTEGER	Queued outgoing message count.	? wsFeed.PendingSends
P04-09 WEBSOCKET.SentMessages INTEGER	Total sent-message count.	? wsFeed.SentMessages
P04-10 WEBSOCKET.ReceivedMessages INTEGER	Total received-message count.	? wsFeed.ReceivedMessages
P04-11 WEBSOCKET.SentBytes INTEGER	Total sent bytes.	? wsFeed.SentBytes
P04-12 WEBSOCKET.ReceivedBytes INTEGER	Total received bytes.	? wsFeed.ReceivedBytes
P04-13 WEBSOCKET.DroppedMessages INTEGER	Dropped incoming message count.	? wsFeed.DroppedMessages
P04-14 WEBSOCKET.ReconnectCount INTEGER	Reconnect-attempt count.	? wsFeed.ReconnectCount
P04-15 WEBSOCKET.Reconnect LOGICAL	Whether automatic reconnection is enabled.	? wsFeed.Reconnect
P04-16 WEBSOCKET.Protocol STRING	Negotiated subprotocol.	? wsFeed.Protocol
P04-17 WEBSOCKET.LastError STRING	Latest WebSocket error.	? wsFeed.LastError
P04-18 WEBSOCKET.LastMessage STRING	Latest received text message.	? wsFeed.LastMessage
P04-19 WEBSOCKET.LastCloseCode INTEGER	Latest close status code.	? wsFeed.LastCloseCode
P04-20 WEBSOCKET.LastCloseReason STRING	Latest close reason.	? wsFeed.LastCloseReason

TASK properties

Property / type	Meaning	Read example
P05-01 TASK.Status STRING	Task state text; do not assume a particular state before waiting.	? task.Status
P05-02 TASK.ThreadId INTEGER	Kokum worker-thread identifier; zero before execution. Multiple tasks can use the same worker.	? task.ThreadId
P05-03 TASK.ErrorMessage	Captured worker error, or an empty string	? task.ErrorMessage

Property / type	Meaning	Read example
STRING	when no error exists.	
P05-04 TASK.IsCompleted LOGICAL	True after successful or failed completion.	? task.IsCompleted
P05-05 TASK.IsRunning LOGICAL	True while the task is pending or running.	? task.IsRunning
P05-06 TASK.IsFaulted LOGICAL	True when the worker ended with an error.	? task.IsFaulted
P05-07 TASK.ElapsedMs INTEGER	Elapsed task time reported in milliseconds.	? task.ElapsedMs

NOTE Declare the receiver with its matching type and initialize it first: task = Double(21), then task.Wait(); see SET-TASK.

MUTEX properties

Property / type	Meaning	Read example
P06-01 MUTEX.Locked LOGICAL	True while the mutex is owned by one Kokum execution.	? gate.Locked
P06-02 MUTEX.OwnerThreadId INTEGER	Logical owner execution identifier, or zero while unlocked.	? gate.OwnerThreadId
P06-03 MUTEX.Waiters INTEGER	Current number of native waiters.	? gate.Waiters
P06-04 MUTEX.Acquisitions INTEGER	Successful lock-acquisition count.	? gate.Acquisitions

NOTE Declare the receiver with its matching type and initialize it first: gate = MUTEX().

ATOMICINTEGER properties

Property / type	Meaning	Read example
P07-01 ATOMICINTEGER.Value INTEGER	Current atomic integer value.	? counter.Value
P07-02 ATOMICINTEGER.Operations INTEGER	Number of atomic operations performed.	? counter.Operations

NOTE Declare the receiver with its matching type and initialize it first: counter = ATOMICINTEGER(5).

CHANNEL properties

Property / type	Meaning	Read example
P08-01 CHANNEL.Capacity INTEGER	Fixed bounded channel capacity.	? queue.Capacity
P08-02 CHANNEL.Count INTEGER	Current number of queued messages.	? queue.Count
P08-03 CHANNEL.Closed LOGICAL	True after the channel has been closed.	? queue.Closed
P08-04 CHANNEL.WaitingSenders INTEGER	Current blocked sender count.	? queue.WaitingSenders
P08-05	Current blocked receiver count.	? queue.WaitingReceivers

Property / type	Meaning	Read example
CHANNEL.WaitingReceivers INTEGER		
P08-06 CHANNEL.Sent INTEGER	Total successfully sent messages.	? queue.Sent
P08-07 CHANNEL.Received INTEGER	Total successfully received messages.	? queue.Received

NOTE Declare the receiver with its matching type and initialize it first: `queue = CHANNEL(2)`.

Common form properties

Property pattern	Small slot example
Text / Enabled / Visible	<code>form.statusBar1.Text = "Ready"</code> <code>form.button1.Enabled = .T.</code>
Geometry	<code>form.button1.X = 20</code> <code>form.button1.Width = 120</code>
CheckBox	<code>form.chkActive.Checked = .T.</code>
ComboBox selection	<code>form.combo1.SelectedIndex = 0</code>
Timer	<code>form.timerPoll.Interval = 250</code> <code>form.timerPoll.Running = .T.</code>
TreeView items / selection	<code>form.tree1.Items = ["Root", " Child A"]</code> <code>form.tree1.SelectedPath = "Root/Child A"</code>
Scroll position	<code>form.scroll1.ScrollY = 300</code>
Splitter	<code>form.splitMain.Position = 240</code>
Table state	? <code>form.table1.RowCount</code> , <code>form.table1.ColumnCount</code>
PropertyGrid state	? <code>form.grid1.PropertyCount</code>

Add the named control before using these examples. The complete schema can be inspected with `kokumc form schema ControlType`. Read-only and runtime-mutable flags are authoritative; a name existing in browser JavaScript does not make it callable from Kokum.

16 / REFERENCE

Coverage, limitations and source provenance

This is documentation of the supplied baseline, not a feature-release announcement.

What was actually checked

Check	Executed scope
Console examples	197 reference examples passed kokumc check, run, run-ir and run-bytecode after correction.
Local HTTP and FlatDB	Six HTTP examples and five Remote FlatDB built-in examples executed against local real Kokum services through all three source execution modes. A two-module project was checked, built, bundled and linked.
GUI/resource examples	112 exact example bodies dispatched through the native KokumVM form/slot bridge: SQLite, graph, image, WebSocket echo, control methods and property alternatives. This is not a browser-layout verification.
Command-line examples	55 noninteractive commands exercised; the uninstalled IDE smoke launch required explicit matching assets/VM paths, reflected in the final example.
Catalogue coverage	105 public built-ins; 112 native methods; 90 native readable properties; 75 language entries; 10 natural patterns; 32 GUI/browser-distinction entries; four HTTP recipes; 78 command-line entries.
Not executed	External TigerDB/PostgreSQL server examples, actual native Windows/macOS applications, full GUI form lifecycle and every interactive CLI scenario. Their required contexts are stated, not silently treated as passing tests.

Important corrections relative to older examples

Older assumption	Current baseline reference
MIN(7,3,9) / MAX(a,b)	Use MIN([7,3,9]) / MAX([a,b]); one ARRAY argument.
REGEXFIND returns a detail MAP	It returns STRING or NULL. Use natural FIND ... WITH DETAILS for maps.
KEXECUTE(sql, values)	Built-in Remote FlatDB accepts one SQL STRING. DATABASE.Execute(sql, values) is a different provider API.
IMPORT ... AS Sum	SUM is a built-in name; the checked module example uses AddTwo.
LOCAL event inside a slot	event is already a parameter. Receive examples use LOCAL message.
TreeView.Items is a STRING in slots	The designer serializes text, but the native slot bridge uses an ARRAY of indented line strings.
Every browser control method is callable	Eleven advertised ScrollView/Splitter/TreeView methods are browser-only here; their entries show alternatives.
HTTPGET / HTTPPOST / EOF	These are not public built-ins in this catalogue. Use URLGET / URLPOST and READLINE(...)=NULL.
CustomCanvas methods	The control is retired in the baseline; do not add it as a new supported API.

Source provenance

Reference	Files within the source archive
[S1] Core language	src/tlang_parser.c; docs/CORE_LANGUAGE_CONTRACT.md; current keyword/type completion lists.

Reference	Files within the source archive
[S2] Collections and values	docs/COLLECTIONS.md; runtime value implementation; array/map regression tests.
[S3] Numeric contracts	docs/DECIMAL_MODEL.md; docs/SCALAR_MATHEMATICS.md; docs/ARRAY_STATISTICS.md.
[S4] Routines, modules and objects	docs/LANGUAGE_MODULES.md; docs/OBJECT_MODEL.md; docs/CODEBLOCKS_AND_CLOSURES.md.
[S5] Concurrency	docs/ASYNC_TASKS.md; named-thread/safe-shared-state phase documentation; src/kokum_task.c and src/kokum_sync.c.
[S6] Catalogue coverage	consolidation/BUILTIN_CATALOG.tsv; src/tlang_builtin_catalog.c; tests/phase4_15_1_2_12/verify_command_catalog.py.
[S7] File and regex APIs	docs/FILE_IO.md; docs/PHASE4_16_1_REGEX_RUNTIME_FOUNDATION.md; src/kokum_regex.c.
[S8] Natural patterns	docs/PHASE4_16_3_NATURAL_FIND_STREAMING.md; docs/PHASE4_16_7_NATURAL_PATTERN_OPERATIONS.md.
[S9] GUI and bindings	src/tlang_gui_schema.c; src/tlang_web_form.c; web/shared/kokum-advanced-widget-contract.json; form and slot examples.
[S10] Databases	docs/SQLITE_PROVIDER.md; docs/TIGERDB_PROVIDER.md; docs/POSTGRES_PROVIDER.md; docs/CREATE_TABLE_FROM_JSON.md; flatdb/.
[S11] Visual resources	docs/GRAPH_WIDGET.md; docs/IMAGE_WIDGET.md; native resource implementations.
[S12] WebSockets	docs/WEBSOCKET_CLIENT.md; src/kokum_websocket.c; native WebSocket/bridge regression fixtures.
[S13] HTTP	docs/URL_HTTP_CLIENT.md; docs/URL_REDIRECT_SECURITY.md; docs/URL_ASYNC_MICROSERVICES_IDE.md; src/kokum_url.c.
[S14] Public tools	kokumc --help; kokumvm --help; kokum-ide --help; current build, bundle, form and backend CLI implementations.

Source archive: Kokum_0_29_0_Baseline_PUBLIC_Preservation_Fix.zip. The base is Kokum 0.29.0 through the Phase 4.17.7.1 MinGW build hotfix with the necessary PUBLIC-preservation IDE correction. Older editions were used for organization only; the source and executed behavior take precedence. No source-code change was made while preparing this book.

Alphabetical index

Command / property · example label · page. Every line is clickable. Browser-only names lead to the explicit distinction and supported alternative.

#

!= [LANG-030].....	5
# (not equal) [LANG-030].....	5
% [LANG-029].....	5
&& [LANG-028].....	5
() grouping [LANG-034].....	6
* [LANG-029].....	5
* comment [LANG-028].....	5
+ [LANG-029].....	5
- [LANG-029].....	5
-- comment [LANG-028].....	5
--headless-smoke [CLI-063].....	96
--no-browser [CLI-060].....	95
--port [CLI-061].....	95
--trace [CLI-062].....	95
. (member access) [LANG-036].....	10
.AND. [LANG-031].....	5
.F. [LANG-022].....	9
.NIL. [LANG-023].....	9
.NOT. [LANG-033].....	6
.NULL. [LANG-023].....	9
.OR. [LANG-032].....	6
.T. [LANG-022].....	9
/ [LANG-029].....	5
// [LANG-028].....	5
:result publication [LANG-063].....	23
; (line continuation) [LANG-027].....	5
< [LANG-030].....	5
<= [LANG-030].....	5
<> [LANG-030].....	5
= (equality) [LANG-030].....	5
== [LANG-030].....	5
> [LANG-030].....	5
>= [LANG-030].....	5
? [LANG-072].....	5
[] indexing [LANG-035].....	10
{ } } [LANG-026].....	9

A

AADD [BI-017].....	36
ABS [BI-005].....	34
ADEL [BI-020].....	37
AINS [BI-019].....	37
AllowCrossOriginBody (URL option) [HTTP-OPTIONS].....	84
AllowInsecureAuth (URL option) [HTTP-OPTIONS].....	84
ALTERNATE [PAT-004].....	51
AND [LANG-031].....	5
ANY type [TYPES].....	10
APPEND [BI-064].....	45
ARRAY literal [LANG-019].....	8
ARRAY type [TYPES].....	10
AS (import alias) [LANG-003].....	19

AS (inheritance) [LANG-052].....	18
AS (type annotation) [LANG-017].....	8
ASET [BI-018].....	36
Assignment (=) [LANG-018].....	8
ASYNC FUNCTION [LANG-008].....	21
Async HTTP [HTTP-004].....	84
ASYNC METHOD [LANG-073].....	22
ASYNC PROCEDURE [LANG-009].....	21
ATOMICADD [BI-084].....	27
ATOMICCOMPAREEXCHANGE [BI-087].....	27
ATOMICDEC [BI-086].....	27
ATOMICGET [BI-082].....	27
ATOMICINC [BI-085].....	27
ATOMICINTEGER [BI-081].....	26
ATOMICINTEGER type [TYPES].....	10
ATOMICINTEGER.Add [API-092].....	31
ATOMICINTEGER.CompareExchange [API-095].....	32
ATOMICINTEGER.Decrement [API-094].....	32
ATOMICINTEGER.Exchange [API-091].....	31
ATOMICINTEGER.Get [API-089].....	31
ATOMICINTEGER.Increment [API-093].....	31
ATOMICINTEGER.Operations (property) [P07-02].....	103
ATOMICINTEGER.Set [API-090].....	31
ATOMICINTEGER.Status [API-096].....	32
ATOMICINTEGER.Value (property) [P07-01].....	103
ATOMICSET [BI-083].....	27
Automatic widget variable [LANG-059].....	54
AVERAGE [BI-054].....	43
AWAIT [LANG-060].....	22

B

BasicAuth (URL option) [HTTP-OPTIONS].....	84
BearerToken (URL option) [HTTP-OPTIONS].....	84
Body (HTTP response) [HTTP-RESPONSE].....	85
BodyType (URL option) [HTTP-OPTIONS].....	84
BOOLEAN type [TYPES].....	10
BREAK [LANG-044].....	13

C

CAFile (URL option) [HTTP-OPTIONS].....	84
CAPTURE [PAT-006].....	51
CASE [LANG-039].....	12
CATCH [LANG-046].....	14
CEIL [BI-042].....	41
CEILING [BI-043].....	41
CHANNEL [BI-088].....	28
CHANNEL type [TYPES].....	10
CHANNEL.Capacity (property) [P08-01].....	103
CHANNEL.Close [API-101].....	33
CHANNEL.Closed (property) [P08-03].....	104
CHANNEL.Count (property) [P08-02].....	103
CHANNEL.Receive [API-099].....	33
CHANNEL.Received (property) [P08-07].....	104
CHANNEL.Send [API-097].....	32
CHANNEL.Sent (property) [P08-06].....	104
CHANNEL.Status [API-102].....	33
CHANNEL.TryReceive [API-100].....	33
CHANNEL.TrySend [API-098].....	32
CHANNEL.WaitingReceivers (property) [P08-05].....	104

DATABASE.Port (property) [P01-06].....	99
DATABASE.Provider (property) [P01-02].....	99
DATABASE.Query [API-011].....	67
DATABASE.QueryOne [API-012].....	67
DATABASE.QueryTable [API-014].....	68
DATABASE.QueryValue [API-013].....	67
DATABASE.Raw [API-008].....	67
DATABASE.ReadOnly (property) [P01-13].....	99
DATABASE.RequestCount (property) [P01-16].....	99
DATABASE.Rollback [API-005].....	66
DATABASE.Schema (property) [P01-10].....	99
DATABASE.ServerVersion (property) [P01-14].....	99
DATABASE.SSLMode (property) [P01-11].....	99
DATABASE.TLS (property) [P01-09].....	99
DATABASE.UseDatabase [API-007].....	66
DATE [BI-010].....	35
DATE type [LANG-025].....	9
DATE type [TYPES].....	10
DATETIME [BI-011].....	35
DATETIME type [LANG-025].....	9
DATETIME type [TYPES].....	10
DB type [TYPES].....	10
DECIMAL [BI-008].....	35
DECIMAL literal [LANG-024].....	9
DECIMAL type [TYPES].....	10
DIGITS [PAT-008].....	51
DISPOSE [LANG-056].....	55
DO CASE [LANG-039].....	12
DO WHILE [LANG-038].....	12

E

EACH [LANG-041].....	13
ElapsedMs (HTTP response) [HTTP-RESPONSE].....	85
ELSE [LANG-037].....	12
ELSEIF [LANG-037].....	12
ENDCASE [LANG-039].....	12
ENDCLASS [LANG-047].....	16
ENDDO [LANG-038].....	12
ENDFUNC [LANG-006].....	15
ENDIF [LANG-037].....	12
ENDITERATE [PAT-009].....	52
ENDLOCK [LANG-069].....	24
ENDMETHOD [LANG-049].....	17
ENDPROC [LANG-007].....	15
ENDTRY [LANG-046].....	14
EQUALS [PAT-002].....	50
EVAL [BI-016].....	36
event [LANG-057].....	53
EXACTLY [PAT-008].....	51
EXCEPT [PAT-004].....	51
EXIT [LANG-043].....	13
EXP [BI-047].....	42
EXPORT [LANG-005].....	20
EXPORT CLASS [LANG-005].....	20
EXPORT FUNCTION [LANG-005].....	20
EXPORT GLOBAL [LANG-005].....	20
EXPORT PROCEDURE [LANG-005].....	20

F

FALSE [LANG-022].....	9
FILE type [TYPES].....	10
FILEHANDLE type [TYPES].....	10
FINALLY [LANG-046].....	14
FIND [PAT-001].....	50
FIRST [PAT-004].....	51
FLOOR [BI-041].....	41
FOLLOWED BY [PAT-006].....	51
FollowRedirects (URL option) [HTTP-OPTIONS].....	84
FOR [LANG-040].....	13
FOR EACH [LANG-041].....	13
FOR EACH index, value [LANG-042].....	13
form [LANG-057].....	53
Form property assignment [LANG-058].....	53
FROM (import) [LANG-003].....	19
FROM FILE [PAT-010].....	52
FROM TEXT [PAT-001].....	50
FUNCTION [LANG-006].....	15

G

GCCOLLECT [BI-033].....	39
GCSTATS [BI-034].....	40
GET requests [BI-105].....	83
GETPROPERTY [BI-031].....	39
GRAPH setup [SET-GRAPH].....	69
GRAPH type [TYPES].....	10
GRAPH.AddCandle [API-031].....	71
GRAPH.AddPoint [API-027].....	71
GRAPH.AddSeries [API-025].....	70
GRAPH.AddSlice [API-029].....	71
GRAPH.Animation (property) [P02-11].....	100
GRAPH.Candles (property) [P02-19].....	101
GRAPH.Clear [API-015].....	69
GRAPH.ClearData [API-016].....	69
GRAPH.EmptyText (property) [P02-10].....	100
GRAPH.GraphType (property) [P02-03].....	100
GRAPH.Id (property) [P02-01].....	100
GRAPH.Labels (property) [P02-17].....	100
GRAPH.LineWidth (property) [P02-14].....	100
GRAPH.MaximumPoints (property) [P02-15].....	100
GRAPH.MaxPoints (property) [P02-16].....	100
GRAPH.Options (property) [P02-20].....	101
GRAPH.Palette (property) [P02-09].....	100
GRAPH.PointCount (property) [P02-23].....	101
GRAPH.Refresh [API-017].....	69
GRAPH.RemoveSeries [API-026].....	70
GRAPH.Revision (property) [P02-21].....	101
GRAPH.Series (property) [P02-18].....	101
GRAPH.SeriesCount (property) [P02-22].....	101
GRAPH.SetAnimation [API-037].....	72
GRAPH.SetCandles [API-030].....	71
GRAPH.SetData [API-021].....	70
GRAPH.SetEmptyText [API-046].....	73
GRAPH.SetGraphType [API-019].....	69
GRAPH.SetGrid [API-035].....	72
GRAPH.SetLabels [API-023].....	70
GRAPH.SetLegend [API-033].....	72
GRAPH.SetLineWidth [API-040].....	73

GRAPH.SetMaximumPoints [API-042].....	73
GRAPH.SetMaxPoints [API-041].....	73
GRAPH.SetOption [API-032].....	71
GRAPH.SetPalette [API-045].....	73
GRAPH.SetPie [API-028].....	71
GRAPH.SetSeries [API-024].....	70
GRAPH.SetSmooth [API-038].....	72
GRAPH.SetStacked [API-039].....	72
GRAPH.SetTitle [API-020].....	70
GRAPH.SetType [API-018].....	69
GRAPH.SetXAxisTitle [API-043].....	73
GRAPH.SetYAxisTitle [API-044].....	73
GRAPH.ShowGrid [API-036].....	72
GRAPH.ShowGrid (property) [P02-06].....	100
GRAPH.ShowLegend [API-034].....	72
GRAPH.ShowLegend (property) [P02-05].....	100
GRAPH.Smooth (property) [P02-12].....	100
GRAPH.Snapshot [API-047].....	74
GRAPH.Stacked (property) [P02-13].....	100
GRAPH.SupplyData [API-022].....	70
GRAPH.Title (property) [P02-04].....	100
GRAPH.Type (property) [P02-02].....	100
GRAPH.XAxisTitle (property) [P02-07].....	100
GRAPH.YAxisTitle (property) [P02-08].....	100
GUI async polling [GUI-TIMER].....	61
GUI control lab [SET-CONTROLS].....	55
GUI slot setup [SET-GUI].....	53

H

HASKEY [BI-022].....	37
HASMETHOD [BI-030].....	39
HASPROPERTY [BI-029].....	39
Headers (HTTP response) [HTTP-RESPONSE].....	85
Headers (URL option) [HTTP-OPTIONS].....	84
HeaderValues (HTTP response) [HTTP-RESPONSE].....	85
HIDE [LANG-054].....	54
HIDEFORM [BI-036].....	55
Historical CLI aliases [CLI-ALIASES].....	98
HTTP options [HTTP-OPTIONS].....	84
HTTP query and headers [HTTP-001].....	83
HTTP response MAP [HTTP-RESPONSE].....	85
HTTP response versus transport error [HTTP-003].....	84
HTTP service setup [SET-HTTP].....	82
HTTP text and form bodies [HTTP-002].....	83
HttpVersion (HTTP response) [HTTP-RESPONSE].....	85

I

IF [LANG-037].....	12
IMAGE setup [SET-IMAGE].....	74
IMAGE type [TYPES].....	10
IMAGE.Clear [API-049].....	74
IMAGE.KeepAspectRatio (property) [P03-02].....	101
IMAGE.Loaded (property) [P03-01].....	101
IMAGE.MimeType (property) [P03-06].....	101
IMAGE.Path (property) [P03-05].....	101
IMAGE.Revision (property) [P03-07].....	101
IMAGE.ScaleMode (property) [P03-03].....	101
IMAGE.SetImage [API-048].....	74
IMAGE.Size (property) [P03-08].....	101

IMAGE.Snapshot [API-050].....	74
IMAGE.Source (property) [P03-04].....	101
IMPORT [LANG-003].....	19
IMPORT BUILTIN [LANG-003].....	19
IMPORT GLOBAL [LANG-003].....	19
IMPORT NATIVE [LANG-003].....	19
IN [LANG-041].....	13
INT [BI-006].....	35
INTEGER type [TYPES].....	10
INTO [PAT-001].....	50
IS BLANK [PAT-008].....	51
IS NOT BLANK [PAT-008].....	51
ITERATE FROM [PAT-009].....	52

J

Json (HTTP response) [HTTP-RESPONSE].....	85
JSON type [TYPES].....	10
JSONDECODE [BI-013].....	36
JSONENCODE [BI-012].....	35
JsonError (HTTP response) [HTTP-RESPONSE].....	85

K

KCONNECT [BI-067].....	63
KCONNECT USING [SET-FLATDB].....	63
KCONNECTED [BI-068].....	64
KDISCONNECT [BI-071].....	64
KEEP DELIMITERS [PAT-007].....	51
KEXECUTE [BI-069].....	64
KEYS [BI-024].....	37
kokum-flatdb-server [CLI-078].....	98
kokum-ide --about [CLI-070].....	97
kokum-ide --assets [CLI-071].....	97
kokum-ide --help [CLI-066].....	96
kokum-ide --kokumvm [CLI-075].....	97
kokum-ide --no-create [CLI-077].....	98
kokum-ide --no-restore [CLI-076].....	98
kokum-ide --no-url [CLI-072].....	97
kokum-ide --recent [CLI-074].....	97
kokum-ide --terminal [CLI-058].....	95
kokum-ide --timeout-ms [CLI-073].....	97
kokum-ide --version [CLI-069].....	97
kokum-ide project [CLI-057].....	95
kokum-ide server [CLI-059].....	95
kokumc --about [CLI-002].....	86
kokumc --help [CLI-064].....	96
kokumc --version [CLI-001].....	86
kokumc ast [CLI-007].....	87
kokumc build [CLI-020].....	89
kokumc build --standalone [CLI-021].....	89
kokumc bundle [CLI-023].....	89
kokumc bundle extract [CLI-038].....	92
kokumc bundle info [CLI-036].....	91
kokumc bundle verify [CLI-037].....	92
kokumc cache add [CLI-039].....	92
kokumc cache list [CLI-040].....	92
kokumc check [CLI-003].....	86
kokumc compile [CLI-004].....	87
kokumc compile-module [CLI-005].....	87
kokumc edit [CLI-043].....	92

kokumc form bind-check [CLI-025].....	90
kokumc form check [CLI-024].....	90
kokumc form compile [CLI-026].....	90
kokumc form disassemble [CLI-029].....	90
kokumc form info [CLI-027].....	90
kokumc form schema [CLI-030].....	91
kokumc form verify [CLI-028].....	90
kokumc ide [CLI-042].....	92
kokumc ir [CLI-009].....	87
kokumc language-server [CLI-041].....	92
kokumc migrate-project [CLI-019].....	89
kokumc package create [CLI-031].....	91
kokumc package extract [CLI-035].....	91
kokumc package info [CLI-032].....	91
kokumc package signing-id [CLI-034].....	91
kokumc package verify [CLI-033].....	91
kokumc project check [CLI-016].....	88
kokumc project graph [CLI-017].....	89
kokumc project info [CLI-015].....	88
kokumc project migrate [CLI-018].....	89
kokumc project new [CLI-014].....	88
kokumc restore [CLI-022].....	89
kokumc run [CLI-011].....	88
kokumc run-bytecode [CLI-013].....	88
kokumc run-ir [CLI-012].....	88
kokumc symbols [CLI-008].....	87
kokumc tokens [CLI-006].....	87
kokumc verify-ir [CLI-010].....	87
kokumc web [CLI-044].....	93
kokumvm --about [CLI-068].....	96
kokumvm --help [CLI-065].....	96
kokumvm --version [CLI-067].....	96
kokumvm disasm [CLI-052].....	94
kokumvm execute bundle [CLI-046].....	93
kokumvm execute bytecode [CLI-045].....	93
kokumvm gui [CLI-049].....	93
kokumvm info [CLI-051].....	94
kokumvm link [CLI-054].....	94
kokumvm link-check [CLI-056].....	94
kokumvm link-entry [CLI-055].....	94
kokumvm profile [CLI-053].....	94
kokumvm serve [CLI-048].....	93
kokumvm verify [CLI-050].....	94
kokumvm web [CLI-047].....	93
KQUERY [BI-070].....	64

L

LAST [PAT-004].....	51
LEN [BI-002].....	34
LETTERS [PAT-008].....	51
LIMIT [PAT-004].....	51
LINE [PAT-001].....	50
LINES [PAT-001].....	50
LOCAL [LANG-013].....	7
LOCK [BI-078].....	26
LOCK block [LANG-069].....	24
LOG [BI-048].....	42
LOG10 [BI-049].....	42
LOGICAL literal [LANG-022].....	9
LOGICAL type [TYPES].....	10

LOOP [LANG-045].....	14
LOWER [BI-004].....	34
LPARAMETERS [LANG-011].....	16

M

Main entry point [SET-MAIN].....	4
MAP literal [LANG-020].....	8
MAP type [TYPES].....	10
MAP.member [LANG-021].....	9
MAPREMOVE [BI-023].....	37
MAPSET [BI-021].....	37
MATCH [PAT-005].....	51
MATCHES [PAT-005].....	51
MATCHES REGEX [PAT-005].....	51
MAX [BI-015].....	36
MaxRedirects (URL option) [HTTP-OPTIONS].....	84
MaxResponseBytes (URL option) [HTTP-OPTIONS].....	84
MEAN [BI-053].....	43
MEDIAN [BI-055].....	43
METHOD [LANG-049].....	17
METHODS [BI-028].....	38
MIN [BI-014].....	36
MOD [BI-050].....	42
MODULE [LANG-001].....	19
Module project setup [SET-MODULE].....	18
MONEY [LANG-024].....	9
MONEY type [TYPES].....	10
Multidimensional arrays [LANG-075].....	10
MUTEX [BI-077].....	26
MUTEX type [TYPES].....	10
MUTEX.Acquisitions (property) [P06-04].....	103
MUTEX.Lock [API-085].....	30
MUTEX.Locked (property) [P06-01].....	103
MUTEX.OwnerThreadId (property) [P06-02].....	103
MUTEX.Status [API-088].....	31
MUTEX.TryLock [API-086].....	30
MUTEX.Unlock [API-087].....	30
MUTEX.Waiters (property) [P06-03].....	103
MUTEXSTATUS [BI-080].....	26

N

Named-thread helper setup [SET-THREAD].....	21
Nested arrays [LANG-075].....	10
NEW [LANG-051].....	18
NEXT [LANG-040].....	13
NIL [LANG-023].....	9
NOT [LANG-033].....	6
NOWAIT [LANG-062].....	22
NULL [LANG-023].....	9
NUMBER type [TYPES].....	10

O

OBJECT type [TYPES].....	10
Ok (HTTP response) [HTTP-RESPONSE].....	85
ON DATA [LANG-070].....	65
ONE OF [PAT-007].....	51
ONE OR MORE [PAT-009].....	52
ONLY [LANG-071].....	65
OPEN [BI-061].....	45

OPTIONAL [LANG-004].....	20
OR [LANG-032].....	6
OTHERWISE [LANG-039].....	12

P

PARAMETERS [LANG-010].....	15
PATTERN type [TYPES].....	10
PERCENTILE [BI-058].....	44
PICTURE type [TYPES].....	10
POST requests [BI-106].....	83
POW [BI-046].....	42
PRIVATE [LANG-014].....	7
PROCEDURE [LANG-007].....	15
PROPERTIES [BI-027].....	38
PROPERTY [LANG-048].....	17
PropertyGrid.AddProperty [WID-028].....	61
PropertyGrid.Clear [WID-032].....	61
PropertyGrid.GetValue [WID-030].....	61
PropertyGrid.RemoveProperty [WID-031].....	61
PropertyGrid.SetValue [WID-029].....	61
PUBLIC [LANG-015].....	7
PUBLIC preservation [SET-GUI].....	53

Q

Query (URL option) [HTTP-OPTIONS].....	84
--	----

R

RANGE [BI-060].....	44
READ [BI-062].....	45
READLINE [BI-066].....	46
ReadTimeoutMs (URL option) [HTTP-OPTIONS].....	84
Reason (HTTP response) [HTTP-RESPONSE].....	85
RedirectCount (HTTP response) [HTTP-RESPONSE].....	85
Reflection setup [SET-REFLECTION].....	38
Regex flags [REGEX-FLAGS].....	49
REGEX type [TYPES].....	10
REGEX.Close [API-112].....	49
REGEX.Find [API-105].....	48
REGEX.FindAll [API-106].....	48
REGEX.Flags [API-110].....	49
REGEX.IsClosed [API-111].....	49
REGEX.IsMatch [API-103].....	48
REGEX.Pattern [API-109].....	49
REGEX.Replace [API-107].....	48
REGEX.Split [API-108].....	49
REGEX.Test [API-104].....	48
REGEX.COMPILE [BI-097].....	46
REGEX.ESCAPE [BI-103].....	47
REGEX.FIND [BI-099].....	47
REGEX.FINDALL [BI-100].....	47
REGEX.MATCH [BI-098].....	46
REGEX.REPLACE [BI-101].....	47
REGEX.SPLIT [BI-102].....	47
Remote FlatDB setup [SET-FLATDB].....	63
REPLACE FROM [PAT-006].....	51
RETURN [LANG-012].....	16
ROUND [BI-040].....	41
RUN ... AS [LANG-061].....	22
RUN ... WAIT FOR [LANG-064].....	23

Running examples [SET-MAIN]..... 4

S

SCHEMA [LANG-071].....	65
SCHEMA ONLY [LANG-071].....	65
ScrollView.EnsureVisible [WID-010].....	57
ScrollView.EnsureVisible (alternative) [WID-010].....	57
ScrollView.ScrollBy [WID-009].....	57
ScrollView.ScrollBy (alternative) [WID-009].....	57
ScrollView.ScrollTo [WID-008].....	57
ScrollX / ScrollY [WID-008].....	57
sender [LANG-057].....	53
SETPROPERTY [BI-032].....	39
SHOW [LANG-053].....	54
SHOWFORM [BI-035].....	55
SIGN [BI-039].....	41
Signal binding [LANG-057].....	53
SLEEP [BI-038].....	24
Slot procedure [LANG-057].....	53
SPLIT FROM [PAT-007].....	51
Splitter.Position [WID-011].....	58
Splitter.ResetSplit [WID-012].....	58
Splitter.ResetSplit (alternative) [WID-012].....	58
Splitter.SetSplitPosition [WID-011].....	58
SQRT [BI-045].....	42
STARTING WITH [PAT-004].....	51
STARTS WITH [PAT-001].....	50
STATIC [LANG-016].....	8
Status (HTTP response) [HTTP-RESPONSE].....	85
STDDEV [BI-057].....	43
STEP [LANG-040].....	13
STR [BI-001].....	34
STRING type [TYPES].....	10
SUM [BI-052].....	43
SYNSTATS [BI-096].....	29

T

TableView.AddColumn [WID-019].....	59
TableView.AddRow [WID-020].....	59
TableView.ClearColumns [WID-025].....	60
TableView.ClearRows [WID-024].....	60
TableView.GetCell [WID-027].....	61
TableView.RemoveColumn [WID-021].....	60
TableView.RemoveRow [WID-022].....	60
TableView.SetCell [WID-026].....	60
TableView.UpdateRow [WID-023].....	60
TASK setup [SET-TASK].....	29
TASK type [TYPES].....	10
TASK.Await [API-082].....	29
TASK.ElapsedMs (property) [P05-07].....	103
TASK.ErrorMessage (property) [P05-03].....	103
TASK.IsCompleted (property) [P05-04].....	103
TASK.IsFaulted (property) [P05-06].....	103
TASK.IsRunning (property) [P05-05].....	103
TASK.Result [API-084].....	30
TASK.Status (property) [P05-01].....	102
TASK.ThreadId (property) [P05-02].....	103
TASK.Wait [API-083].....	30
THIS [LANG-050].....	17

THREADDONE [BI-073].....	25
THREADERROR [BI-074].....	25
THREADID [BI-075].....	25
THREADPOOLSIZE [BI-076].....	25
THREADSTATUS [BI-072].....	25
TIMEOUT [LANG-068].....	24
TimeoutMs (URL option) [HTTP-OPTIONS].....	84
Timer.timerTick [GUI-TIMER].....	61
TLSServerName (URL option) [HTTP-OPTIONS].....	84
TO [LANG-040].....	13
Toolchain lab setup [SET-TOOLS].....	86
TreeView.AddItem [WID-013].....	58
TreeView.Clear [WID-015].....	58
TreeView.Clear (alternative) [WID-015].....	58
TreeView.Collapse [WID-017].....	59
TreeView.Collapse (alternative) [WID-017].....	59
TreeView.Expand [WID-016].....	59
TreeView.Expand (alternative) [WID-016].....	59
TreeView.Items [WID-013].....	58
TreeView.RemoveItem [WID-014].....	58
TreeView.RemoveItem (alternative) [WID-014].....	58
TreeView.Select [WID-018].....	59
TreeView.SelectedPath [WID-018].....	59
TRUE [LANG-022].....	9
TRUNC [BI-044].....	42
TRY [LANG-046].....	14
Type names and aliases [TYPES].....	10
Typed declarations and aliases [LANG-074].....	10
TYPEOF [BI-009].....	35

U

UNLOCK [BI-079].....	26
UPPER [BI-003].....	34
Url (HTTP response) [HTTP-RESPONSE].....	85
URLGET [BI-105].....	83
URLPOST [BI-106].....	83
UserAgent (URL option) [HTTP-OPTIONS].....	84

V

VAL [BI-007].....	35
VALIDATE FROM [PAT-008].....	51
VALUES [BI-025].....	38
VARIANCE [BI-056].....	43
VerifyTLS (URL option) [HTTP-OPTIONS].....	84
VERSION [LANG-002].....	19
VOID type [TYPES].....	10

W

WAIT FOR [LANG-065].....	23
WAIT FOR ALL [LANG-067].....	24
WAIT FOR list [LANG-066].....	23
WEBSOCKET setup [SET-WEBSOCKET].....	75
WEBSOCKET type [TYPES].....	10
WEBSOCKET.AddQueryParameter [API-063].....	78
WEBSOCKET.ClearHeaders [API-056].....	77
WEBSOCKET.ClearMessages [API-080].....	81
WEBSOCKET.ClearQueryParameters [API-065].....	78
WEBSOCKET.Close [API-068].....	79
WEBSOCKET.Connect [API-067].....	79

WEBSOCKET.Connected (property) [P04-05].....	102
WEBSOCKET.Disconnect [API-069].....	79
WEBSOCKET.Drain [API-079].....	81
WEBSOCKET.DroppedMessages (property) [P04-13].....	102
WEBSOCKET.Id (property) [P04-01].....	101
WEBSOCKET.IsConnected (property) [P04-04].....	101
WEBSOCKET.IsRunning (property) [P04-06].....	102
WEBSOCKET.LastCloseCode (property) [P04-19].....	102
WEBSOCKET.LastCloseReason (property) [P04-20].....	102
WEBSOCKET.LastError (property) [P04-17].....	102
WEBSOCKET.LastMessage (property) [P04-18].....	102
WEBSOCKET.PendingMessages (property) [P04-07].....	102
WEBSOCKET.PendingSends (property) [P04-08].....	102
WEBSOCKET.Ping [API-073].....	80
WEBSOCKET.Poll [API-076].....	80
WEBSOCKET.Protocol (property) [P04-16].....	102
WEBSOCKET.Receive [API-075].....	80
WEBSOCKET.ReceivedBytes (property) [P04-12].....	102
WEBSOCKET.ReceivedMessages (property) [P04-10].....	102
WEBSOCKET.ReceiveJSON [API-078].....	81
WEBSOCKET.ReceiveText [API-077].....	80
WEBSOCKET.Reconnect (property) [P04-15].....	102
WEBSOCKET.ReconnectCount (property) [P04-14].....	102
WEBSOCKET.RemoveHeader [API-055].....	77
WEBSOCKET.RemoveQueryParameter [API-064].....	78
WEBSOCKET.SendBinary [API-072].....	79
WEBSOCKET.SendJSON [API-071].....	79
WEBSOCKET.SendText [API-070].....	79
WEBSOCKET.SentBytes (property) [P04-11].....	102
WEBSOCKET.SentMessages (property) [P04-09].....	102
WEBSOCKET.SetApiKey [API-059].....	77
WEBSOCKET.SetBasicAuth [API-060].....	78
WEBSOCKET.SetBearerToken [API-057].....	77
WEBSOCKET.SetCookie [API-061].....	78
WEBSOCKET.SetFeedToken [API-058].....	77
WEBSOCKET.SetHeader [API-054].....	77
WEBSOCKET.SetProtocols [API-052].....	76
WEBSOCKET.SetQueryParameter [API-062].....	78
WEBSOCKET.SetReconnect [API-066].....	78
WEBSOCKET.SetSubprotocols [API-053].....	77
WEBSOCKET.SetUrl [API-051].....	76
WEBSOCKET.Snapshot [API-081].....	81
WEBSOCKET.State (property) [P04-03].....	101
WEBSOCKET.Url (property) [P04-02].....	101
WEBSOCKET.Wait [API-074].....	80
WHILE [LANG-038].....	12
WITH DETAILS [PAT-005].....	51
WITH INDEX [PAT-009].....	52
WITH replacement [PAT-006].....	51
WORD [PAT-002].....	50
WORDS [PAT-002].....	50
WRITE [BI-063].....	45
WSCLIENT type [TYPES].....	10