webside

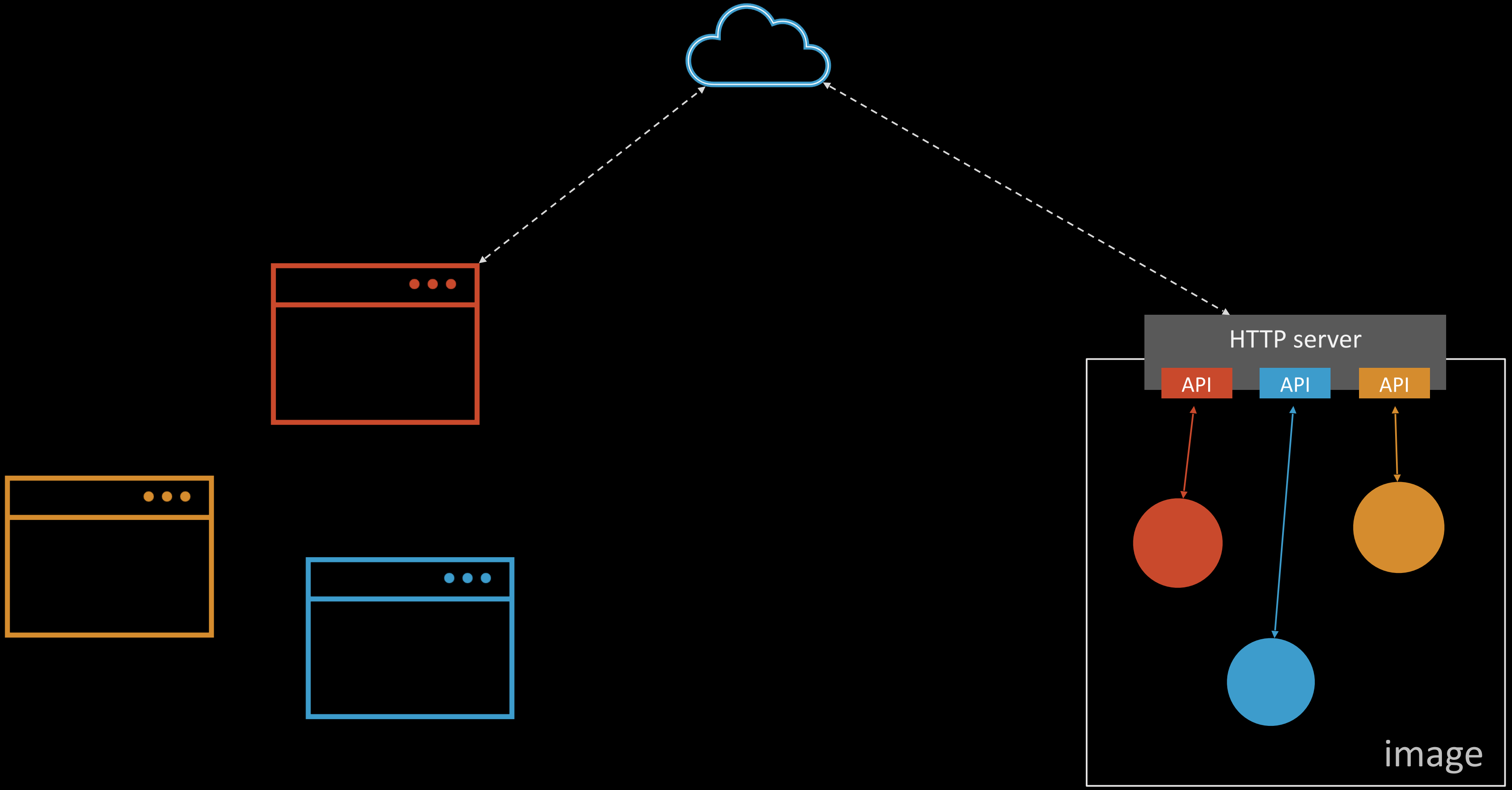
guille amaral

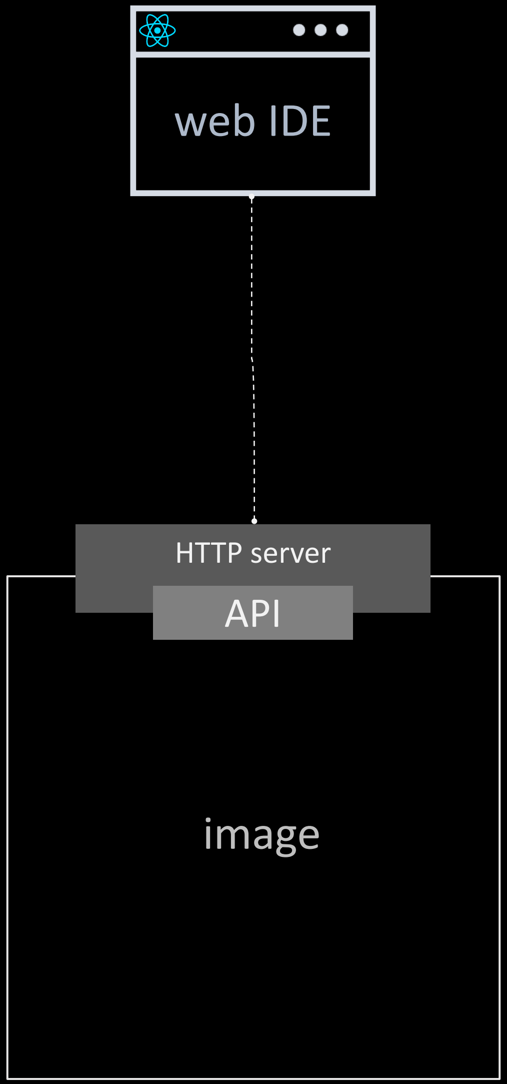
quorum
software

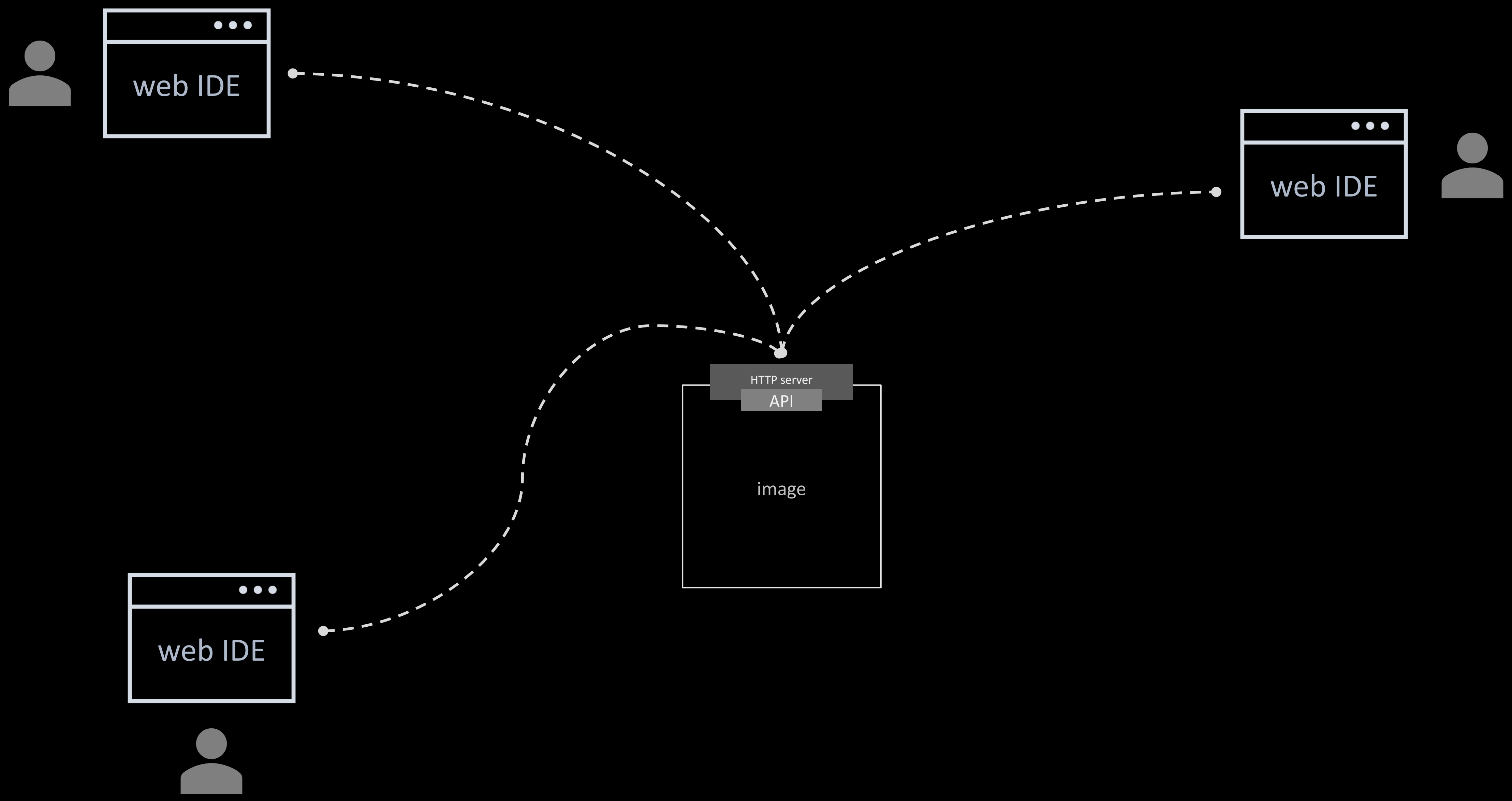
webside

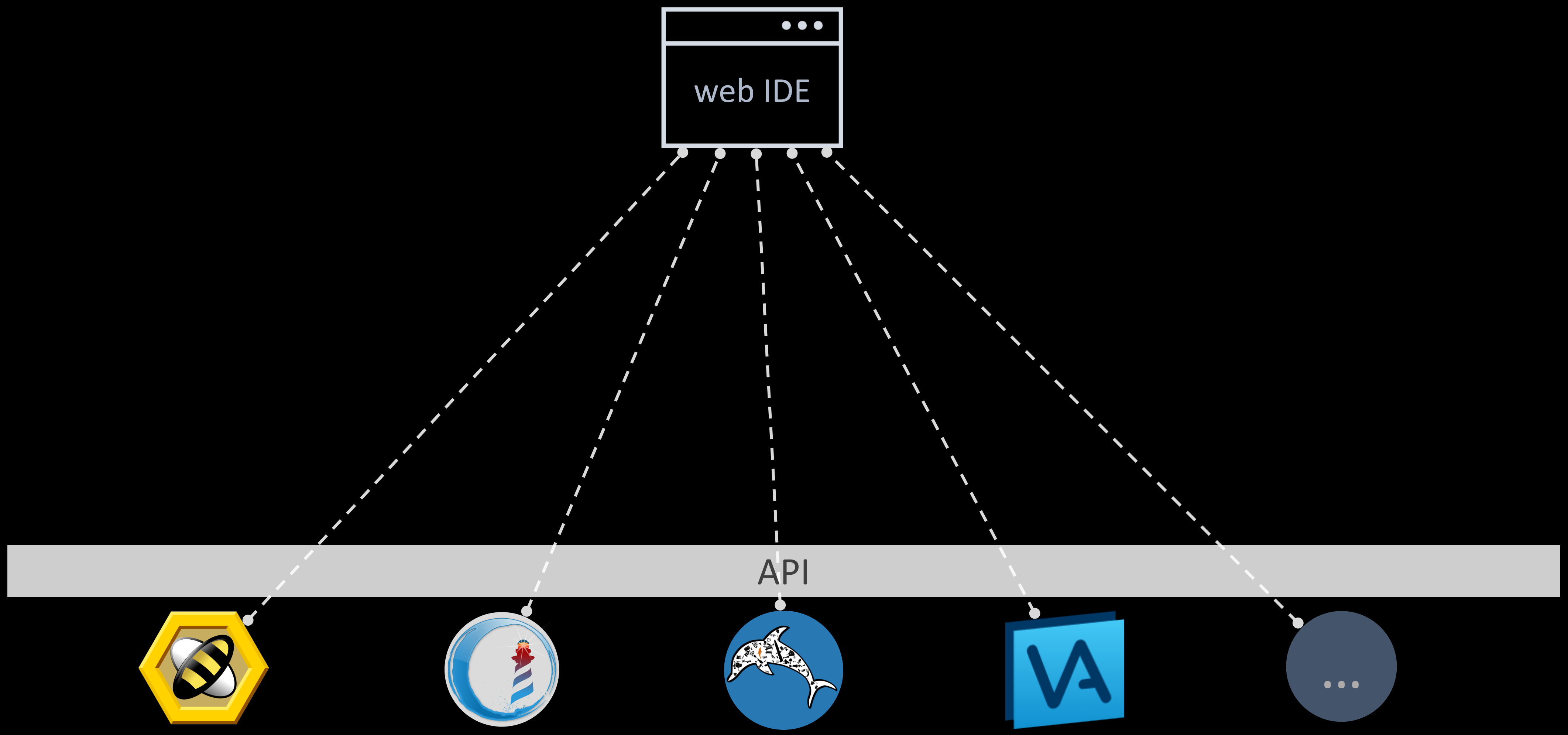
web IDE

React


HTTP server
API
API
API
image


web IDE
HTTP server
API
image


web IDE
web IDE
web IDE
HTTP server
API
image



web IDE

API

Face/Off (1997)

API

code

changes

evaluations / processes

objects

debuggers

workspaces

test runs

profilers

image

API

code

changes

evaluations / processes

objects

debuggers

workspaces

test runs

profilers

image

```
classes
    /classes
        /classes?root=<classname>                               //hierarchy
    /classes/<name>
variables
    /classes/<name>/instance-variables
    /classes/<name>/class-variables
    /classes/<name>/variables
categories
    /classes/<name>/categories
methods
    /classes/<name>/methods
    /classes/<name>/methods/<selector>
    /methods
        /methods?sending=<selector>                             //senders
        /methods?selector=<selector>                            //implementors
        /methods?referencing=<global>                           //references
        /methods?sending=<selector>&referencing=<global>…        //any combination
```

GET /classes/Fraction

200 - OK

```
{
    "name": "Fraction",
    "definition": "Number subclass: #Fraction\r\tinstanceVariableNames: 'numerator denominator'\r\tclassVariableNames: ''\r\tpoolDic
tionaries: ''\r\tcategory: 'Kernel-Numbers'",
    "superclass": "Number",
    "comment": "Fraction provides methods for dealing with fractions like 1/3 as fractions (not as 0.33333...).  All public arithmet
ic operations answer reduced fractions (see examples).
\r\rinstance variables: 'numerator denominator '\r\rExamples: (note the parentheses required to get the right answers in Smalltalk a
nd Pharo):
\r\r(2/3) + (2/3)\r(2/3) + (1/2)\t\t \"answers shows the reduced fraction\" \r(2/3) raisedToInteger: 5\t\t \"fractions also can have
 exponents\"\r",
    "package": "Kernel"
}
```

GET /classes?root=Boolean

200 - OK

```
[
  {
        "name": "Boolean",
        "definition": "Object subclass: #Boolean\r\tinstanceVariableNames: ''\r\tclassVariableNames: ''\r\tpoolDictionaries: ''\r\tcategory: 'Ker
nel-Objects'",
        "superclass": "Object",
        "comment": "Boolean is an abstract class defining the protocol for logic testing operations and conditional control structures for the lo
gical values represented by the instances of its subclasses True and False.
\r\rBoolean redefines #new so no instances of Boolean can be created. It also redefines several messages in the 'copying' protocol to ensure that
 only one instance of each of its subclasses True (the global true, logical assertion) and False (the global false, logical negation) ever exist
in the system.",
        "package": "Kernel"
    },
  {
        "name": "False",
        "definition": "Boolean subclass: #False\r\tinstanceVariableNames: ''\r\tclassVariableNames: ''\r\tpoolDictionaries: ''\r\tcategory: 'Kern
el-Objects'",
        "superclass": "Boolean",
        "comment": "False defines the behavior of its single instance, false -- logical negation. Notice how the truth-
value checks become direct message sends, without the need for explicit testing.
\r\rBe aware however that most of these methods are not sent as real messages in normal use. Most are inline coded by the compiler as test and ju
mp bytecodes - avoiding the overhead of the full message sends. So simply redefining these methods here will have no effect.",
        "package": "Kernel"
    },
    {
        "name": "True",
        "definition": "Boolean subclass: #True\r\tinstanceVariableNames: ''\r\tclassVariableNames: ''\r\tpoolDictionaries: ''\r\tcategory: 'Kerne
l-Objects'",
        "superclass": "Boolean",
        "comment": "True defines the behavior of its single instance, true -- logical assertion. Notice how the truth-
value checks become direct message sends, without the need for explicit testing.
\r\rBe aware however that most of these methods are not sent as real messages in normal use. Most are inline coded by the compiler as test and ju
mp bytecodes - avoiding the overhead of the full message sends. So simply redefining these methods here will have no effect.",
        "variable": false,
        "package": "Kernel"
    },
]
```

GET /classes/Fraction/methods

200 - OK

```
[
    {
        "selector": "*",
        "methodClass": "Fraction",
        "category": "arithmetic",
        "package": "Kernel",
        "source": "* aNumber \r\t\"Answer the result of multiplying the receiver by aNumber.\"\r\t| d1 d2 |
\r\taNumber isFraction ifTrue: \r\t\t[d1 := numerator gcd: aNumber denominator.
\r\t\td2 := denominator gcd: aNumber numerator.\r\t\t(d2 = denominator and: [d1 = aNumber denominator])
\r\t\t\tifTrue: [^ numerator // d1 * (aNumber numerator // d2)].
\r\t\t^ Fraction numerator: numerator // d1 * (aNumber numerator // d2)\r\t\t\t\tdenominator: denominator // d2 * (aNumbe
r denominator // d1)].\r\t^ aNumber adaptToFraction: self andSend: #*"
    },
    {
        "selector": "+",
        "methodClass": "Fraction",
        "category": "arithmetic",
        "package": "Kernel",
        "source": "+ aNumber \r\t\"Answer the sum of the receiver and aNumber.\"\r\t| n d d1 d2 |
\r\taNumber isInteger ifTrue:\r\t\t[^Fraction numerator: numerator + (denominator * aNumber) denominator: denominator].
\r\taNumber isFraction ifTrue: \r\t\t[d := denominator gcd: aNumber denominator.
\r\t\tn := numerator * (d1 := aNumber denominator // d) + (aNumber numerator * (d2 := denominator // d)).
\r\t\td1 := d1 * d2.\r\t\tn := n // (d2 := n gcd: d).\r\t\t(d := d1 * (d // d2)) = 1 ifTrue: [^ n].
\r\t\t^ Fraction numerator: n denominator: d].\r\t^ aNumber adaptToFraction: self andSend: #+"
    },
…
]
```

GET /classes/Fraction/methods/truncated

200 - OK

```
{
    "selector": "truncated",
    "methodClass": "Fraction",
    "category": "truncation and round off",
    "package": "Kernel",
    "source": "truncated \r\t\"Refer to the comment in Number|truncated.\"\r\r\t^numerator quo: denominator",
}
```

GET /classes/Fraction/variables

200 - OK

```
[
    {
        "name": "numerator",
        "class": "Fraction",
        "type": "instance"
    },
    {
        "name": "denominator",
        "class": "Fraction",
        "type": "instance"
    },
    {
        "name": "DependentsFields",
        "class": "Object class",
        "type": "class"
    }
]
```

GET /classes/Fraction/categories

200 - OK

```
[
    "arithmetic",
    "self evaluating",
    "converting",
    "printing",
    "*Math-Operations-Extensions",
    "truncation and round off",
    "private",
    "comparing",
    "*STON-Core",
    "testing"
]
```

GET /methods?selector=reciprocal

200 - OK

```
[
  …
  {
        "selector": "reciprocal",
        "methodClass": "Fraction",
        "category": "private",
        "package": "Kernel",
        "source": "reciprocal\r\t\r\tnumerator abs = 1 ifTrue: [^denominator * numerator].
\r\t^self class numerator: denominator denominator: numerator"
    },
  {
        "selector": "reciprocal",
        "methodClass": "Number",
        "category": "arithmetic",
        "package": "Kernel",
        "source": "reciprocal\r\t\"Returns the reciprocal of self.
\r\tIn case self is 0 the / signals ZeroDivide\"\r\t\r\t\"1/2 reciprocal >>> 2\"\r\t\"2 reciprocal >>> (1/2)\"\r\t\"1.25
reciprocal >>> 0.8\"\r\t\"-2 reciprocal >>> (-1/2)\"\r\t\r\t^1 / self"
    },
  {
        "selector": "reciprocal",
        "methodClass": "Point",
        "category": "arithmetic",
        "package": "Kernel",
        "source": "reciprocal\r\t\"Answer a Point with coordinates that are the reciprocals of mine.
\"\r\t\"(100@200) reciprocal >>> ((1/100)@(1/200))\"\r\t\r\t^ x reciprocal @ y reciprocal"
    },
  …
]
```

API

code

changes

evaluations / processes

objects

debuggers

workspaces

test runs

profilers

image

changes

/classes

/classes/<name>

/changes

/classes/<name>/instance-variables

/classes/<name>/class-variables

/classes/<name>/categories

/classes/<name>/methods

/methods

…

```
"AddMethod"
"RemoveMethod"
"ClassifyMethod"
"RenameMethod"
"AddClass"
"CommentClass"
"RemoveClass"
"RenameClass"
"AddInstanceVariable"
"RemoveInstanceVariable"
"RenameInstanceVariable"
"MoveUpInstanceVariable"
"MoveDownInstanceVariable"
"AddClassVariable"
"RemoveClassVariable"
"RenameClassVariable"
"RenameCategory"
"RemoveCategory"
"AddPackage"
"RemovePackage"
"RenamePackage“
…
```

POST /changes

```
{
    "type": "AddMethod",
    "package": "Kernel",
    "author": "guille",
    "class": "Fraction",
    "sourceCode": "blah: aNumber\r\t^self class numerator: numerator + aNumber denominator: denominator - aNumber",
    "category": "arithmetic"
}
```

200 - OK

```
{
    "type": "AddMethod",
    "label": "Fraction>>blah:",
    "package": “Kernel",
    "timestamp": "2022-08-17T07:25:37.761-03:00",
    "author": "guille",
    "class": "Fraction",
    "sourceCode": "blah: aNumber\r\t^self class numerator: numerator + aNumber denominator: denominator - aNumber",
    "selector": "blah:",
    "category": "arithmetic"
}
```

POST /changes

```
{
    "type": "AddMethod",
    "author": "guille",
    "class": "Object",
    "package": "Kernel",
    "category": "associating",
    "sourceCode": "blah\r\t^1 +"
}
```

409 - Conflict

```
{
    "description": "Variable or expression expected",
    "interval": {
        "start": 11,
        "end": 11
    }
}
```

POST /changes

```
{
    "type": "AddMethod",
    "author": "guille",
    "class": "Object",
    "package": "Kernel",
    "category": "associating",
    "sourceCode": "blah\r\t^x + 1"
}
```

409 - Conflict

```
{
    "description": "Undeclared variable: #x ",
    "interval": {
        "start": 8,
        "end": 8
    },
    "suggestions": [
        {
            "description": "Declare #x as temporary variable",
            "changes": [
                {
                    "type": "AddMethod",
                    "author": "guille",
                    "class": "Object",
                    "sourceCode": "blah\r\t | x | \r\t^x + 1",
                    "category": "associating"
                }
            ]
        },
        {
            "description": "Declare #x as instance variable",
            "changes": [
                {
                    "type": "AddInstanceVariable",
                    "author": "guille",
                    "class": "Object",
                    "variable": "x"
                },
                {
                    "type": "AddMethod",
                    "author": "guille",
                    "class": "Object",
                    "sourceCode": "blah\r\t^x + 1",
                    "category": "associating"
                }
            ]
        }
    ]
}
```

API

code

changes

evaluations / processes

objects

debuggers

workspaces

test runs

profilers

image

POST /evaluations

```
{
    "expression": "1 @ 2 extent: 10 @ 10",
    "sync": true,                                          //synchronous
    "pin": false
}
```

200 - OK

```
{
    "id": “33",
    "class": "Rectangle",
    "indexable": false,
    "printString": "1 @ 2 rightBottom: 11 @ 12"
}
```

POST /evaluations

```
{
    "expression": "1 @ 2 extent: 10 @ 10",
    "sync": false,
    "pin": false
}
```

//asynchronous

200 - OK

```
{
    "id": “41",
    "expression": "1 @ 2 extent: 10 @ 10",
    "state": "pending"
}
```

API

code

changes

evaluations / processes

objects

debuggers

workspaces

test runs

profilers

image

GET /objects/41

//synchronous

200 - OK

```
{
  "id": “41",
  "class": "Rectangle",
  "indexable": false,
  "printString": "1 @ 2 rightBottom: 11 @ 12"
}
```

GET /objects

200 - OK

```
[
  …
  {
    "id": “33",
    "class": "Rectangle",
    "indexable": false,
    "printString": "1 @ 2 rightBottom: 11 @ 12"
  }
  …
]
```

GET /classes/Rectangle/instance-variables

200 - OK

```
[
    {
        "name": "origin",
        "class": "Rectangle",
        "type": "instance"
    },
    {
        "name": "corner",
        "class": "Rectangle",
        "type": "instance"
    }
]
```

GET /objects/41/corner

200 - OK

```
{
  "class": "Point",
  "printString": "11 @ 12"
}
```

API

code

changes

evaluations / processes

objects

debuggers

workspaces

test runs

profilers

image

POST /evaluations

```
{
    "expression": "123 blah",
    "sync": true,
    "pin": false
}
```

409 - Conflict

```
{
    "description": "Instance of SmallInteger did not understand #blah",
    "evaluation": "53"
}
```

POST /debuggers

```
{
    "evaluation": "53"
}
```

200 - OK

```
{
    "id": "77",
    "description": "Instance of SmallInteger did not understand #blah"
}
```

GET /debuggers/77/frames

200 - OK

```
[
    {
        "index": 1,
        "label": "UndefinedObject>>#DoIt"
    },
    {
        "index": 2,
        "label": "a CompiledBlock: [ receiver withArgs: (context ifNil: [ #() ] ifNotNil: [ {context} ]) executeMethod: self compileDoit]"
    },
    {
        "index": 3,
        "label": "BlockClosure>>#on:do:"
    },
    …
]
```

POST /debuggers/77/frames/3/restart

200 - OK

GET /debuggers/77/frames

200 - OK

```
[
    {
        "index": 1,
        "label": "BlockClosure>>#on:do:"
    },
    {
        "index": 2,
        "label": "OpalCompiler>>#evaluate"
    },
    {
        "index": 3,
        "label": "OpalCompiler>>#evaluate:"
    },
    {
        "index": 4,
        "label": "WebsideAPI>>#evaluateExpression:"
    },
    {
        "index": 5,
        "label": "a CompiledBlock: [ object := self evaluateExpression: expression ]"
    },
    {
        "index": 6,
        "label": "BlockClosure>>#on:do:"
    },
   …
]
```

GET /debuggers/77/frames/1/bindings

200 - OK

```
[
    {
        "name": "self",
        "value": "[ receiver withArgs: (context ifNil: [ #() ] ifNotNil: [ {context} ]) executeMethod: self compileDoit]"
    },
    {
        "name": "exception",
        "value": "SyntaxErrorNotification"
    },
    {
        "name": "handlerAction",
        "value": "[ :exception |
 \r\t\t\tself compilationContext requestor\r                ifNotNil: [\r\t\t\t\t\t\tself compilationContext requestor \r\t\t\t\t\t\t\tnotify:
exception errorMessage , ' ->'\r\t\t\t\t\t\t\tat: exception location\r\t\t\t\t\t\t\tin: exception errorCode.
\r                    self compilationContext failBlock value ]\r                ifNil: [ exception pass ]]"
    },
    {
        "name": "outerContext",
        "value": "OpalCompiler>>evaluate"
    },
    {
        "name": "startpc",
        "value": "a CompiledBlock: [ receiver withArgs: (context ifNil: [ #() ] ifNotNil: [ {context} ]) executeMethod: self compileDoit]"
    },
   …
]
```

API

code

changes

evaluations / processes

objects

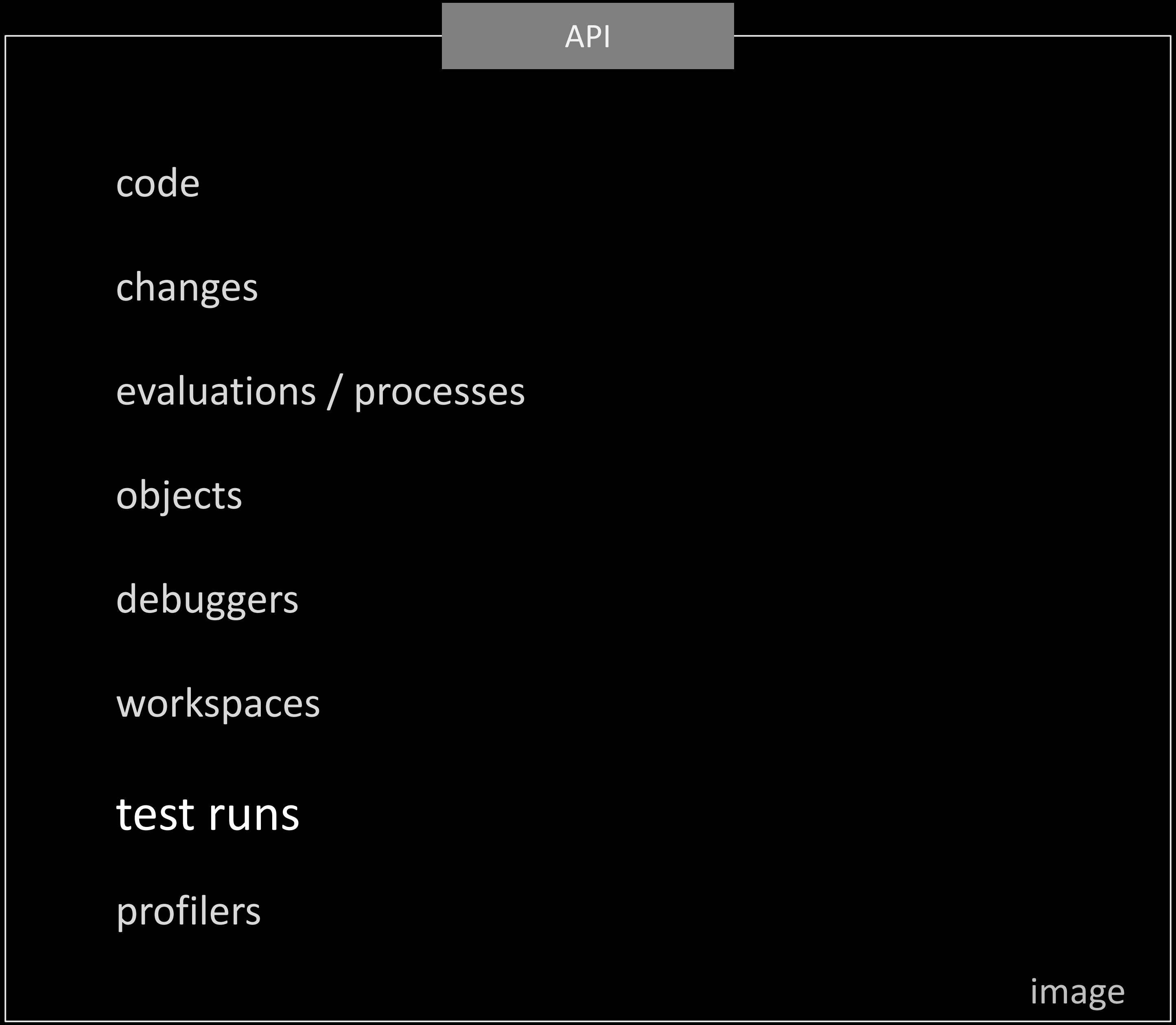
debuggers

workspaces

test runs

profilers

image



API

code

changes

evaluations / processes

objects

debuggers

workspaces

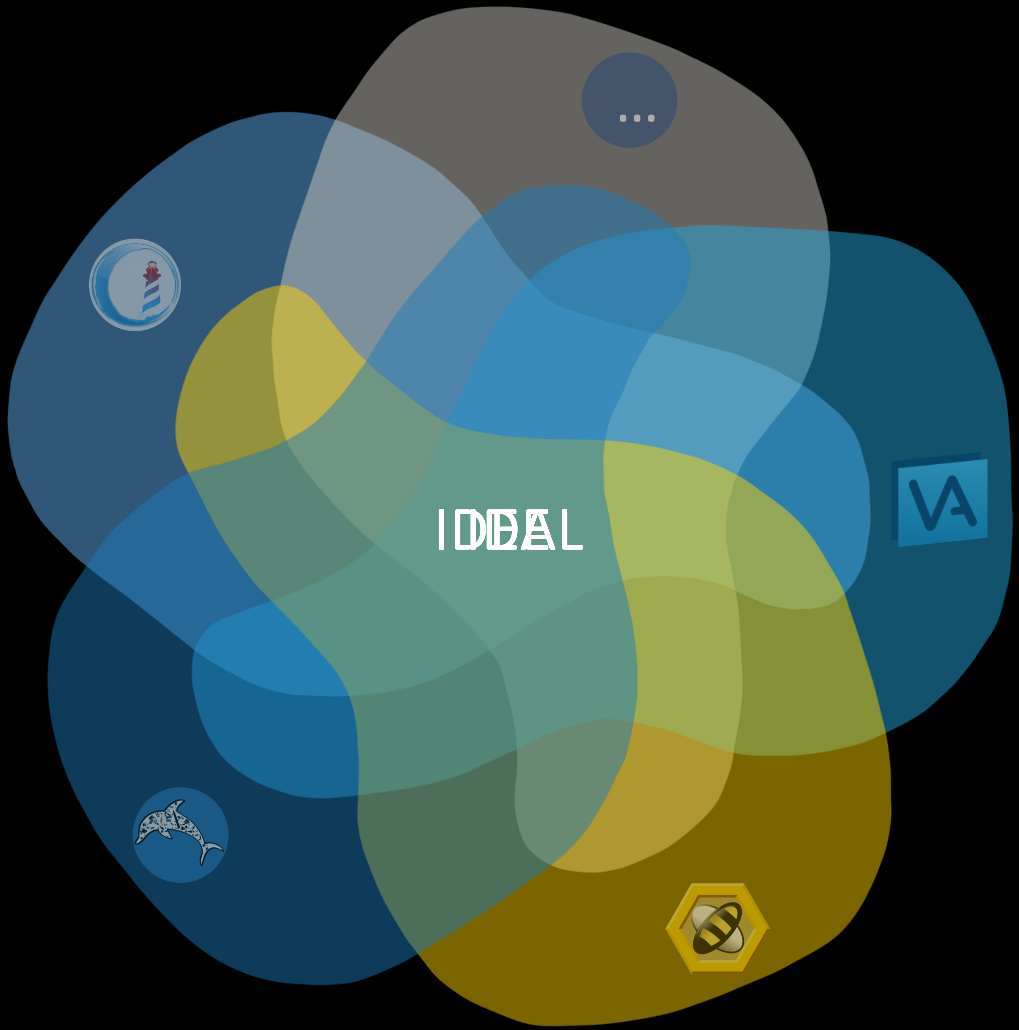
test runs

profilers

image

challenge

- common set of operations
  - is it worth it?



IDEAL

DBA

https://github.com/guillermoamaral/Pharo-Webside

https://github.com/guillermoamaral/Bee-Webside

https://github.com/guillermoamaral/Dolphin-Webside

https://github.com/
guillermoamaral/VAST-Webside

cons

- IDE not inspectable/extensible using the same language
- no true concurrency
- messages with UI effects do not reach the front-end
- impossible common subset of options
- server can be a bottleneck
- no cache cleaning policy (pinned objects, running processes, etc.)

pros

- fully operational IDE
- well documented API
- thin API implementation
  - e.g., couple classes of ~75 methods each, plus ~80 extension methods in Pharo
- tested and used for real (Bee experience)
- first IDE option for a new dialect/runtime
  - e.g., Powerlang (J. Pimas)
- might be useful in some cases

cases

- remote access (e.g., container)
- code migration
- modest collaborative environment
- comparing implementation/performance between dialects

takeaways & next steps

- super fun and interesting tour by several dialects
- API implementation may not be the best (except the Bee one) / dialect experts can improve it!
- still buggy: PR / contributions welcome
- next station: GemStone

thanks

questions