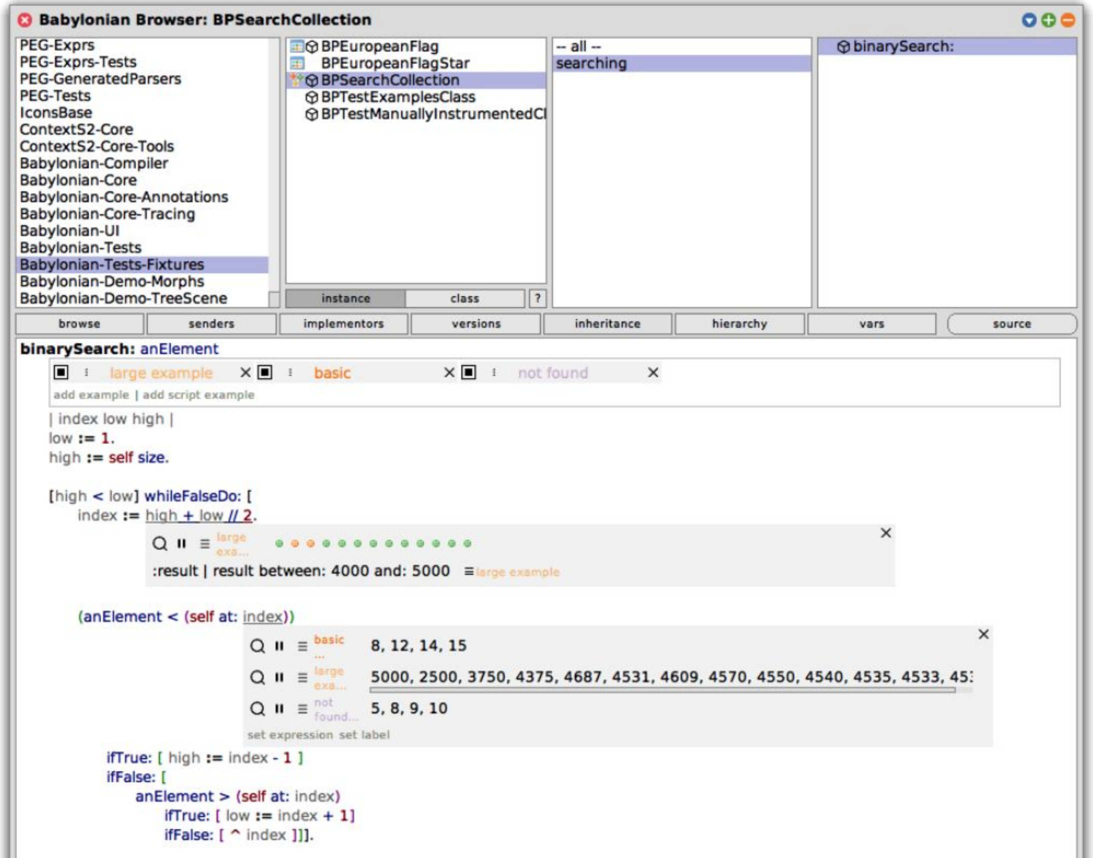
Babylonian in Intellij

Live Programming (Sommersemester 2021) - Paul Methfessel, 22.07.2021

Slides: https://docs.google.com/presentation/d/19nYB_HyBzF-UrK3cD46-vBtas0VApnmvzUaoDQaU6tg/edit?usp=sharing
Repository: https://github.com/Paulpanther/intellij-babylonian-plugin

Interactive Code Inlays - Babylonian/S



PEG-Exprs
PEG-Exprs-Tests
PEG-GeneratedParsers
PEG-Tests
IconsBase
ContextS2-Core
ContextS2-Core-Tools
Babylonian-Compiler
Babylonian-Core
Babylonian-Core-Annotations
Babylonian-Core-Tracing
Babylonian-UI
Babylonian-Tests
Babylonian-Tests-Fixtures
Babylonian-Demo-Morphs
Babylonian-Demo-TreeScene

BPEuropeanFlag
BPEuropeanFlagStar
BPSearchCollection
BPTestExamplesClass
BPTestManuallyInstrumentedCl

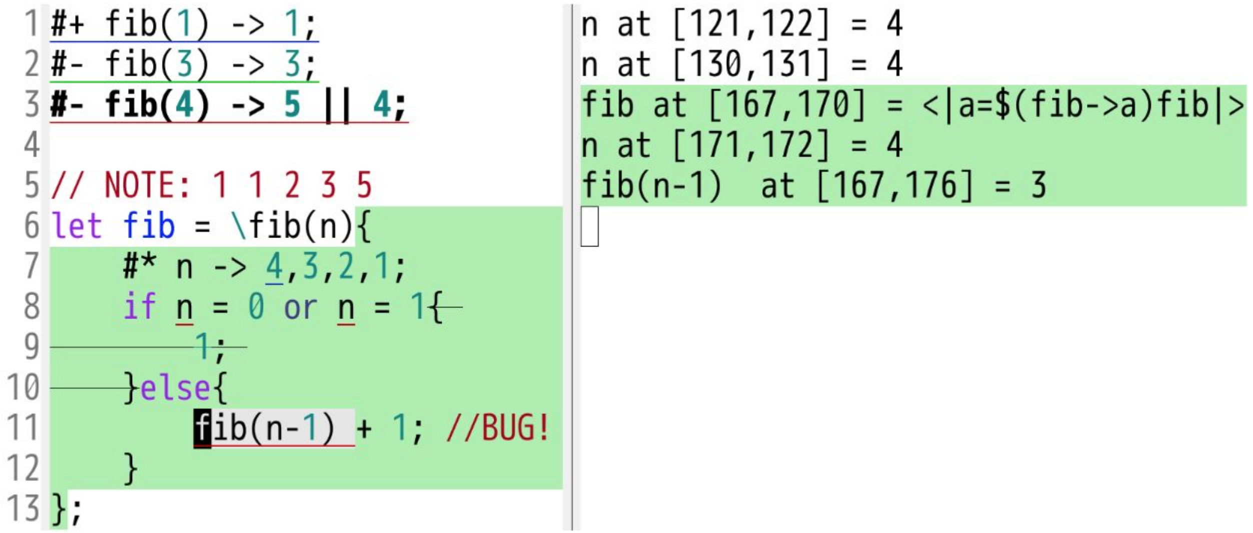
-- all --
searching

binarySearch:

```
binarySearch: anElement
    large example ×   basic ×   not found ×
    | index low high |
    low := 1.
    high := self size.

    [high < low] whileFalseDo: [
        index := high + low // 2.
                large exa...
                :result | result between: 4000 and: 5000   large example

        (anElement < (self at: index))
                basic ...       8, 12, 14, 15
                large exa...    5000, 2500, 3750, 4375, 4687, 4531, 4609, 4570, 4550, 4540, 4535, 4533, 45:
                not found...    5, 8, 9, 10
            ifTrue: [ high := index - 1 ]
            ifFalse: [
                anElement > (self at: index)
                    ifTrue: [ low := index + 1]
                    ifFalse: [ ^ index ]]].
```

https://github.com/hpi-swa-lab/
babylonian-programming-smalltalk

Two-Pane Layout - Shiranui

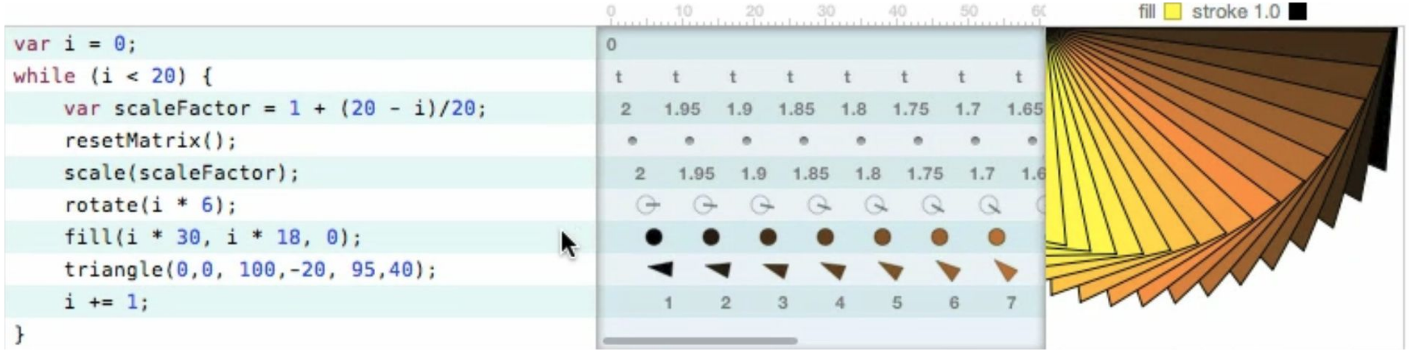


```
#+ fib(1) -> 1;
#- fib(3) -> 3;
#- fib(4) -> 5 || 4;

// NOTE: 1 1 2 3 5
let fib = \fib(n){
    #* n -> 4,3,2,1;
    if n = 0 or n = 1{
        1;
    }else{
        fib(n-1) + 1; //BUG!
    }
};
```

```
n at [121,122] = 4
n at [130,131] = 4
fib at [167,170] = <|a=$(fib->a)fib|>
n at [171,172] = 4
fib(n-1)  at [167,176] = 3
```

Tomoki Imai, Hidehiko Masuhara, and Tomoyuki Aotani. “Shiranui: A Live Programming with Support for Unit Testing”. In: Companion Proceedings of the ACM SIGPLAN International Conference on Systems, Programming, Languages and Applications: Software for Humanity (SPLASH) 2015. Pittsburgh, PA, USA: ACM, 2015, pages 36–37. isbn: 978-1-4503-3722-9. doi: 10.1145/2814189.2817268.

Live-Specimen - Learnable Programming



```
var i = 0;
while (i < 20) {
    var scaleFactor = 1 + (20 - i)/20;
    resetMatrix();
    scale(scaleFactor);
    rotate(i * 6);
    fill(i * 30, i * 18, 0);
    triangle(0,0, 100,-20, 95,40);
    i += 1;
}
```

```
0
  t     t     t     t     t     t     t     t
  2     1.95  1.9   1.85  1.8   1.75  1.7   1.65
  2     1.95  1.9   1.85  1.8   1.75  1.7   1.6
        1     2     3     4     5     6     7
```

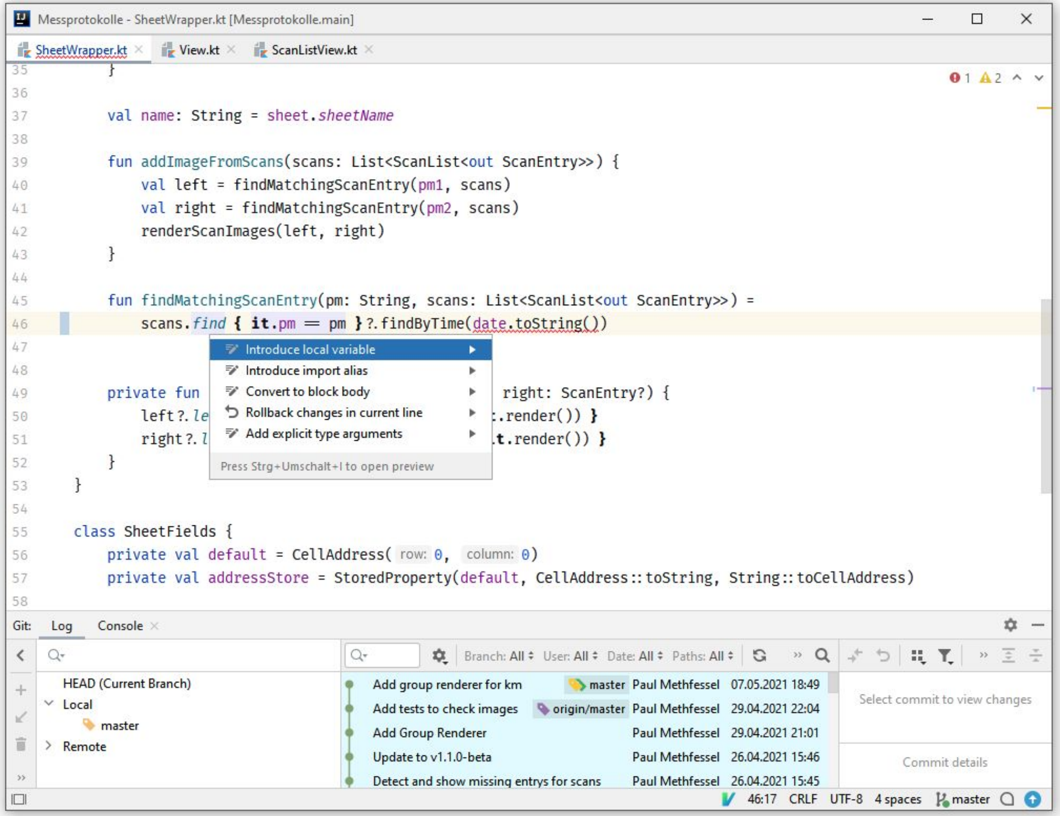
Question:

How can we integrate ELP into traditional IDEs?

Limitation:

Only small set of graphical concepts (because of discoverable interaction)

Intellij IDEA



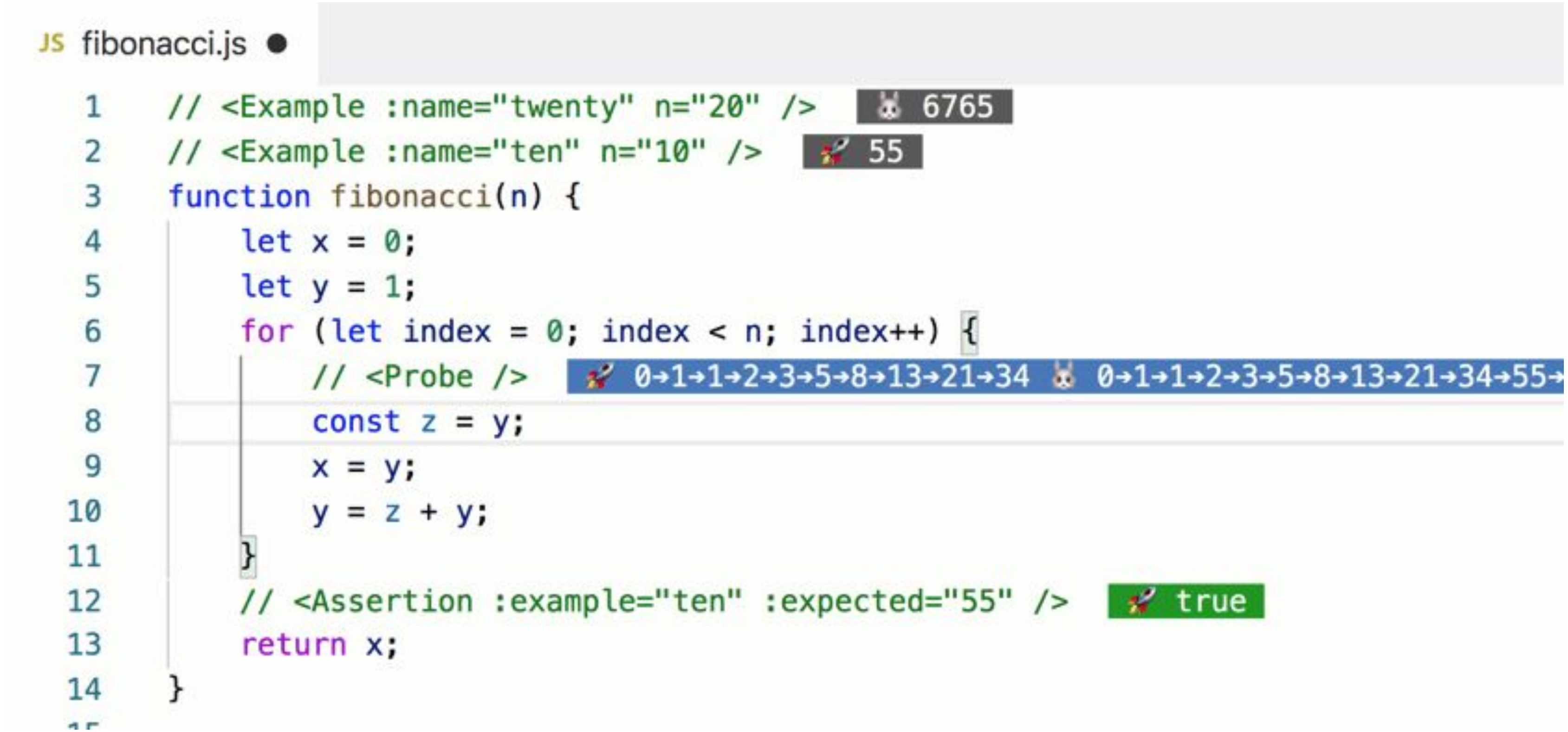
Messprotokolle - SheetWrapper.kt [Messprotokolle.main]

SheetWrapper.kt × View.kt × ScanListView.kt ×

```
        }

        val name: String = sheet.sheetName

        fun addImageFromScans(scans: List<ScanList<out ScanEntry>>) {
            val left = findMatchingScanEntry(pm1, scans)
            val right = findMatchingScanEntry(pm2, scans)
            renderScanImages(left, right)
        }

        fun findMatchingScanEntry(pm: String, scans: List<ScanList<out ScanEntry>>) =
            scans.find { it.pm == pm }?.findByTime(date.toString())


        private fun                                      right: ScanEntry?) {
            left?.le                                   :.render()) }
            right?.l                                   .t.render()) }
        }
    }

    class SheetFields {
        private val default = CellAddress( row: 0,  column: 0)
        private val addressStore = StoredProperty(default, CellAddress::toString, String::toCellAddress)
```

Introduce local variable
Introduce import alias
Convert to block body
Rollback changes in current line
Add explicit type arguments
Press Strg+Umschalt+I to open preview
1
2
Git: Log Console
Branch: All User: All Date: All Paths: All
HEAD (Current Branch)
Local
master
Remote
Select commit to view changes
Commit details
46:17 CRLF UTF-8 4 spaces master

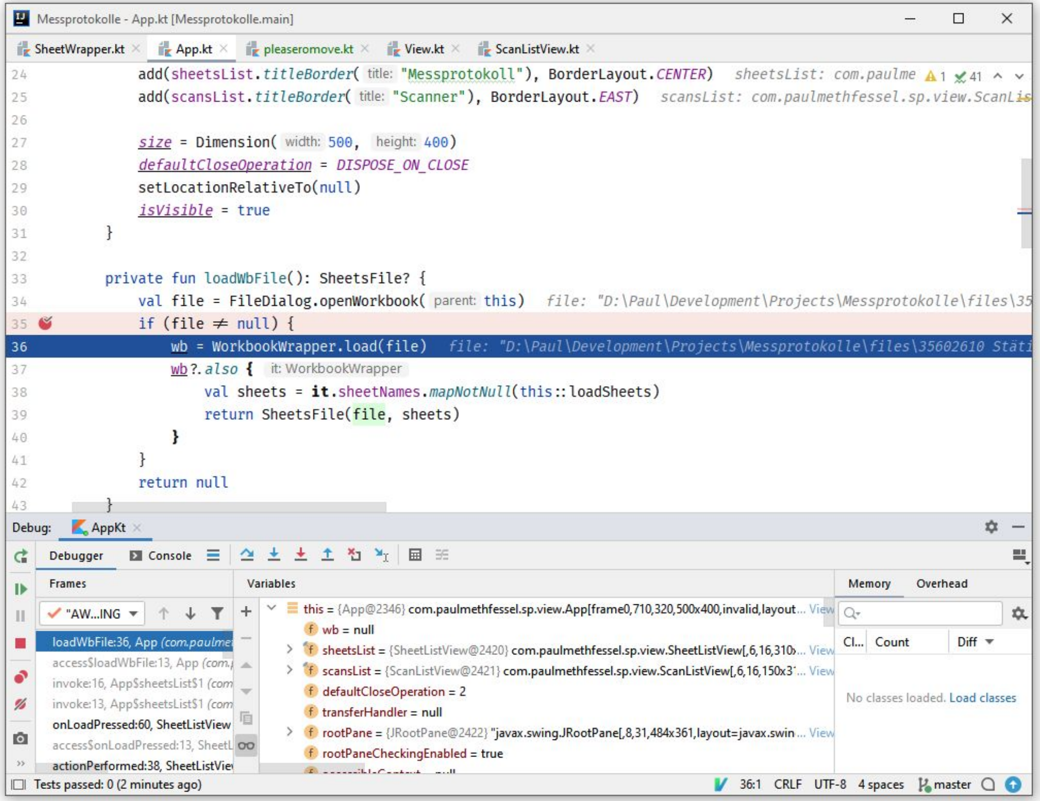
```
Add group renderer for km             master         Paul Methfessel  07.05.2021 18:49
Add tests to check images             origin/master  Paul Methfessel  29.04.2021 22:04
Add Group Renderer                                   Paul Methfessel  29.04.2021 21:01
Update to v1.1.0-beta                                Paul Methfessel  26.04.2021 15:46
Detect and show missing entrys for scans             Paul Methfessel  26.04.2021 15:45
```



JS fibonacci.js ●

```
// <Example :name="twenty" n="20" />   🐰 6765
// <Example :name="ten" n="10" />   🚀 55
function fibonacci(n) {
    let x = 0;
    let y = 1;
    for (let index = 0; index < n; index++) {
        // <Probe />   🚀 0→1→1→2→3→5→8→13→21→34 🐰 0→1→1→2→3→5→8→13→21→34→55→
        const z = y;
        x = y;
        y = z + y;
    }
    // <Assertion :example="ten" :expected="55" />   🚀 true
    return x;
}
```

Intellij IDEA


Messprotokolle - App.kt [Messprotokolle.main]

SheetWrapper.kt · App.kt · pleaseromove.kt · View.kt · ScanListView.kt

```
        add(sheetsList.titleBorder( title: "Messprotokoll"), BorderLayout.CENTER)   sheetsList: com.paulme
        add(scansList.titleBorder( title: "Scanner"), BorderLayout.EAST)   scansList: com.paulmethfessel.sp.view.ScanLis

        size = Dimension( width: 500,  height: 400)
        defaultCloseOperation = DISPOSE_ON_CLOSE
        setLocationRelativeTo(null)
        isVisible = true
    }

    private fun loadWbFile(): SheetsFile? {
        val file = FileDialog.openWorkbook( parent: this)   file: "D:\Paul\Development\Projects\Messprotokolle\files\35
        if (file ≠ null) {
            wb = WorkbookWrapper.load(file)   file: "D:\Paul\Development\Projects\Messprotokolle\files\35602610 Stäti
            wb?.also {  it: WorkbookWrapper
                val sheets = it.sheetNames.mapNotNull(this::loadSheets)
                return SheetsFile(file, sheets)
            }
        }
        return null
    }
```

Debug: AppKt

Debugger · Console

Frames

```
"AW...ING
loadWbFile:36, App (com.paulme
access$loadWbFile:13, App (com.
invoke:16, App$sheetsList$1 (com
invoke:13, App$sheetsList$1 (com
onLoadPressed:60, SheetListView
access$onLoadPressed:13, SheetL
actionPerformed:38, SheetListVie
```

Variables

```
this = {App@2346} com.paulmethfessel.sp.view.App[frame0,710,320,500x400,invalid,layout... View
    wb = null
  > sheetsList = {SheetListView@2420} com.paulmethfessel.sp.view.SheetListView[,6,16,310... View
  > scansList = {ScanListView@2421} com.paulmethfessel.sp.view.ScanListView[,6,16,150x3... View
    defaultCloseOperation = 2
    transferHandler = null
  > rootPane = {JRootPane@2422} "javax.swing.JRootPane[,8,31,484x361,layout=javax.swin... View
    rootPaneCheckingEnabled = true
```

Memory · Overhead

Cl...	Count	Diff

No classes loaded. Load classes

Tests passed: 0 (2 minutes ago)   36:1   CRLF   UTF-8   4 spaces   master

<demo>

Shown Features

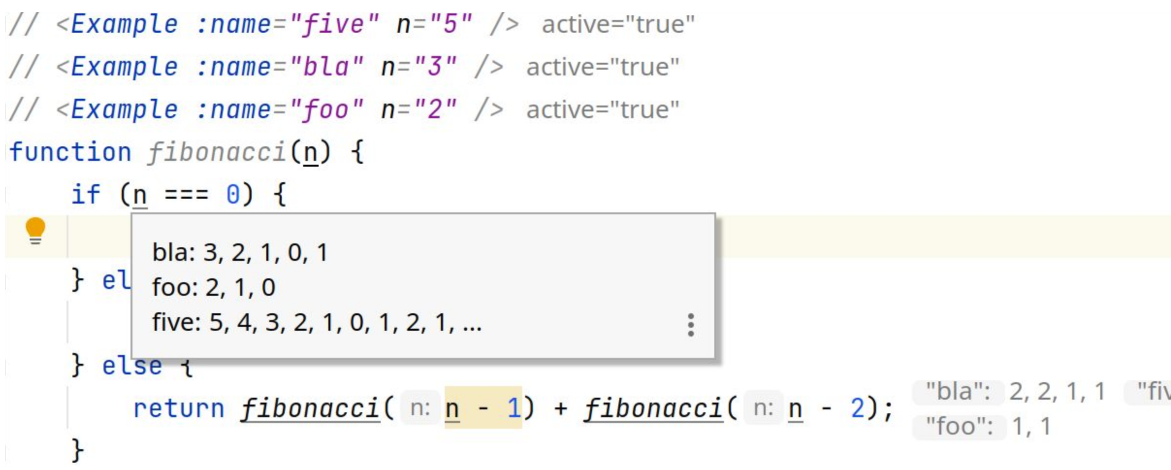
Examples in Comments

Probes via Hover

Probes via Selection

Locked Probes

Active/Inactive Examples



```
// <Example :name="five" n="5" />  active="true"
// <Example :name="bla" n="3" />  active="true"
// <Example :name="foo" n="2" />  active="true"
function fibonacci(n) {
    if (n === 0) {

    } el
        
    } else {
        return fibonacci( n: n - 1) + fibonacci( n: n - 2);   "bla": 2, 2, 1, 1   "fiv
                                                               "foo": 1, 1
    }
```

```
bla: 3, 2, 1, 0, 1
foo: 2, 1, 0
five: 5, 4, 3, 2, 1, 0, 1, 2, 1, ...
```

Mapping

Babylonian/S

- Interactive Example Inlays
- Interactive Probe Inlays
- Create Probes through selection and menu

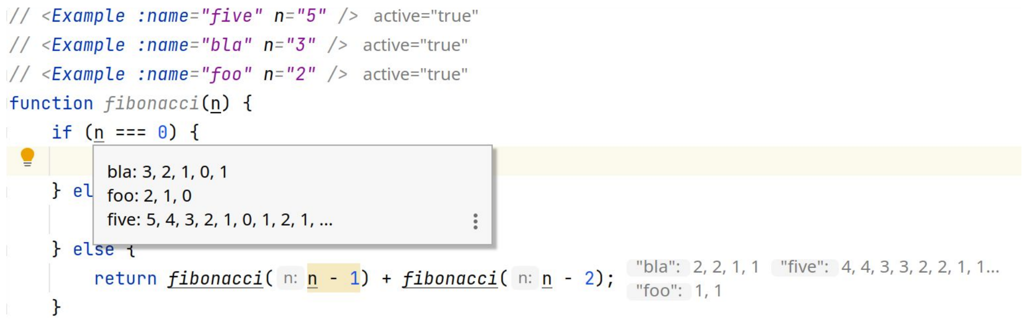
Intellij Plugin

- Examples in Code (Interactivity through menu)
- Probe Inlays (No Interactivity)
- Create Probes through selection
- No Live Specimen

Design Decisions

Examples are in Code, Probes not

- Examples are relevant for other programmers (synced via VCS)
- Non-Code Probes can be set (and removed) by the IDE
- Examples have human-readable information (helpful even without plugin)



```
// <Example :name="five" n="5" />  active="true"
// <Example :name="bla" n="3" />  active="true"
// <Example :name="foo" n="2" />  active="true"
function fibonacci(n) {
    if (n === 0) {

    } el

    } else {
        return fibonacci( n: n - 1) + fibonacci( n: n - 2);  "bla": 2, 2, 1, 1  "five": 4, 4, 3, 3, 2, 2, 1, 1...
                                                             "foo": 1, 1
    }
```

```
bla: 3, 2, 1, 0, 1
foo: 2, 1, 0
five: 5, 4, 3, 2, 1, 0, 1, 2, 1, ...
```

Design Decisions

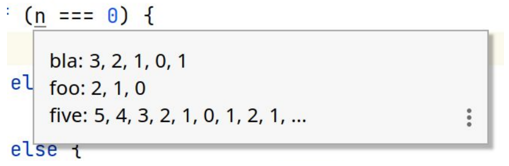
Probes are refreshed on save

- (Because Babylonian/S does this)
- Refreshing on Keystroke is too slow and may execute on unfinished code
- Disadvantage: Probes show old/wrong values when not saved yet

Design Decisions

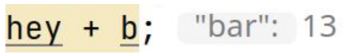
Probes can be seen with hover/selection

- Allows very fast insight
- Is less cluttered than showing all possible probe values
- Disadvantage: Will not show “surprising” values



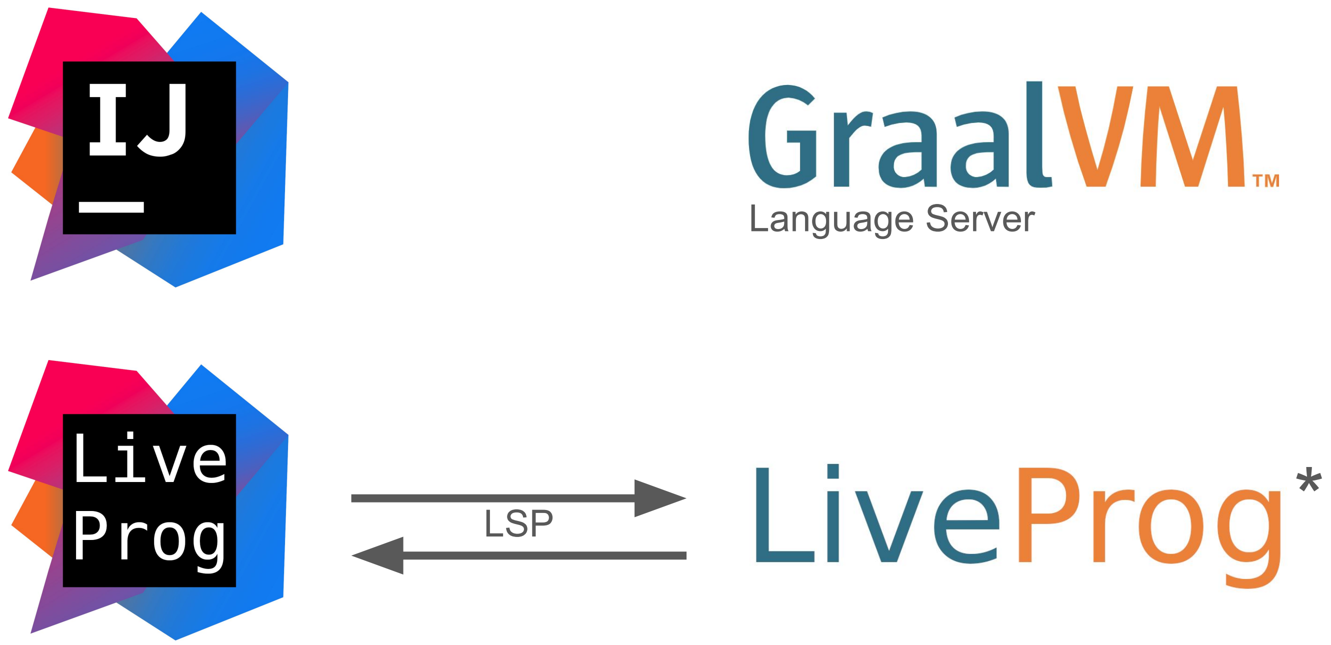
```
(n === 0) {
el
else {
```

bla: 3, 2, 1, 0, 1
foo: 2, 1, 0
five: 5, 4, 3, 2, 1, 0, 1, 2, 1, ...



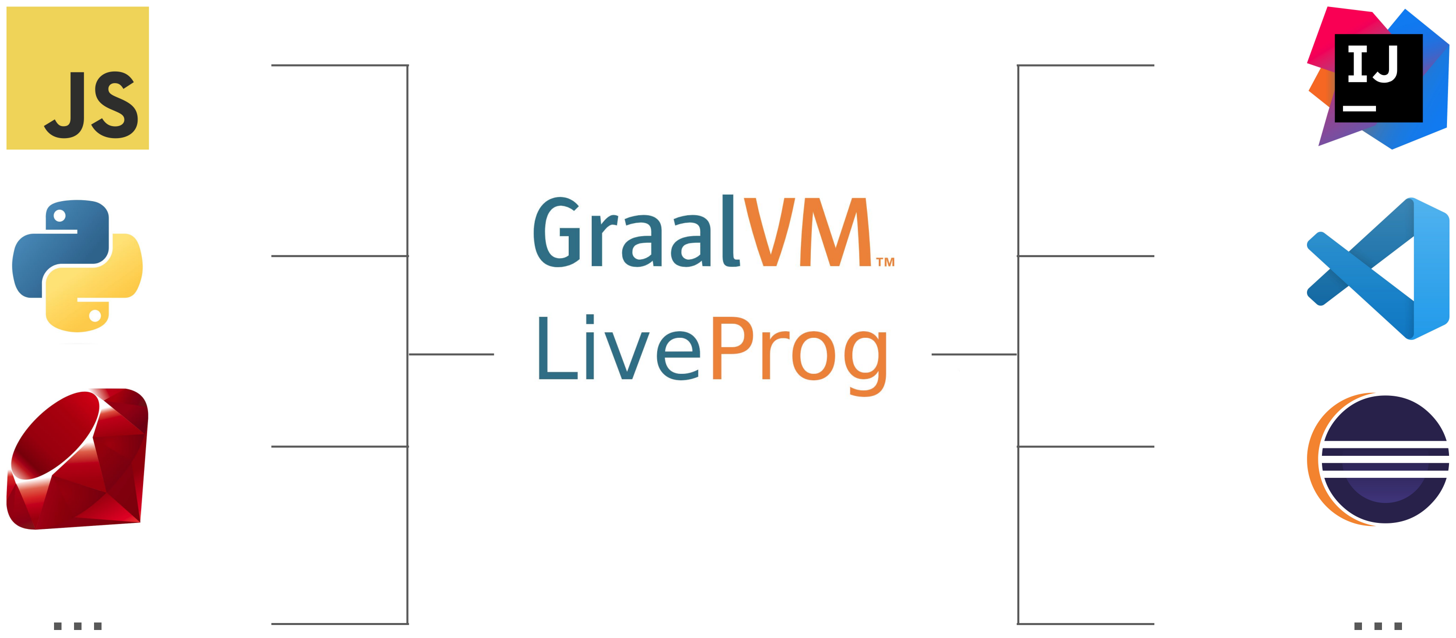
```
hey + b;  "bar": 13
```

Technology Overview



* https://github.com/hpi-swa/polyglot-live-programming

GraalVM Extension



GraalVM Extension

```
fun babylonian_analysis(
    file: URL,
    probePoints: List<FileRange>,
    activeExamples: List<ActiveExample>
): List<Probe>
```

```
class Probe(
    val position: FileRange,
    val examples: List<ProbeExample>
)
```

```
class ProbeExample(
    val name: String,
    val observedValues: List<String>
)
```

(simplified code)

GraalVM Extension

Existing features

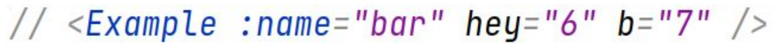
- Scanning file for examples/probes
- Analyzing Probe values

Newly added features

- Find Probe by text-range and not only line
- Activate/Deactivate Examples

Intellij Plugin

- **Annotator**
- Inlay Hint
- Line Marker
- Intention Action
- Documentation Provider

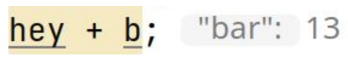


```
// <Example :name="bar" hey="6" b="7" />
```

Docs: plugins.jetbrains.com/docs/intellij

Intellij Plugin

- Annotator
- **Inlay Hint**
- Line Marker
- Intention Action
- Documentation Provider

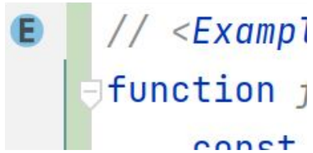


```
hey + b;  "bar": 13
```

Docs: plugins.jetbrains.com/docs/intellij

Intellij Plugin

- Annotator
- Inlay Hint
- **Line Marker**
- Intention Action
- Documentation Provider

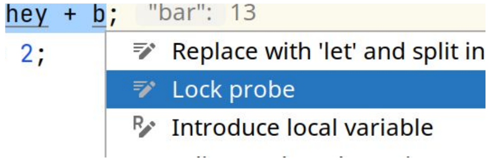


```
// <Exampl
function j
    const
```

Docs: plugins.jetbrains.com/docs/intellij

Intellij Plugin

- Annotator
- Inlay Hint
- Line Marker
- **Intention Action**
- Documentation Provider

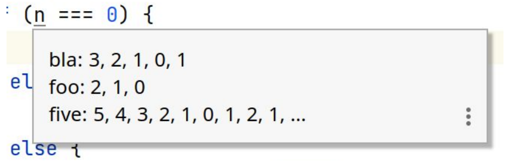


```
hey + b;  "bar": 13
2;
```

Docs: plugins.jetbrains.com/docs/intellij

Intellij Plugin

- Annotator
- Inlay Hint
- Line Marker
- Intention Action
- **Documentation Provider**

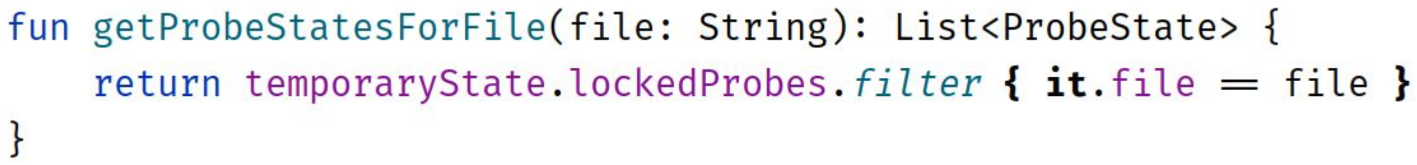


```
(n === 0) {
el
else {
```

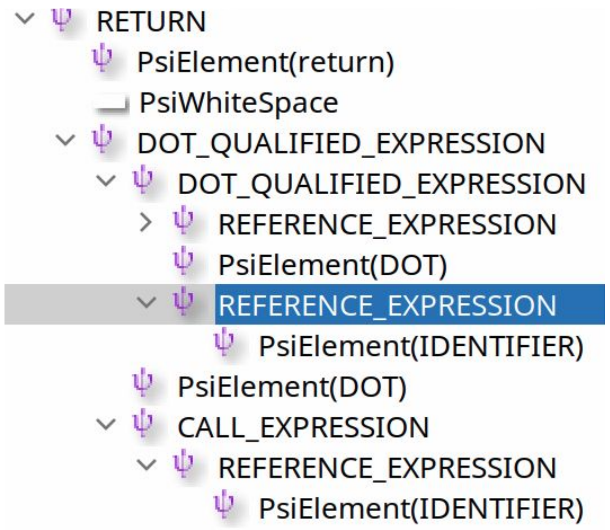
bla: 3, 2, 1, 0, 1
foo: 2, 1, 0
five: 5, 4, 3, 2, 1, 0, 1, 2, 1, ...

Docs: plugins.jetbrains.com/docs/intellij

Intellij Plugin - PSI

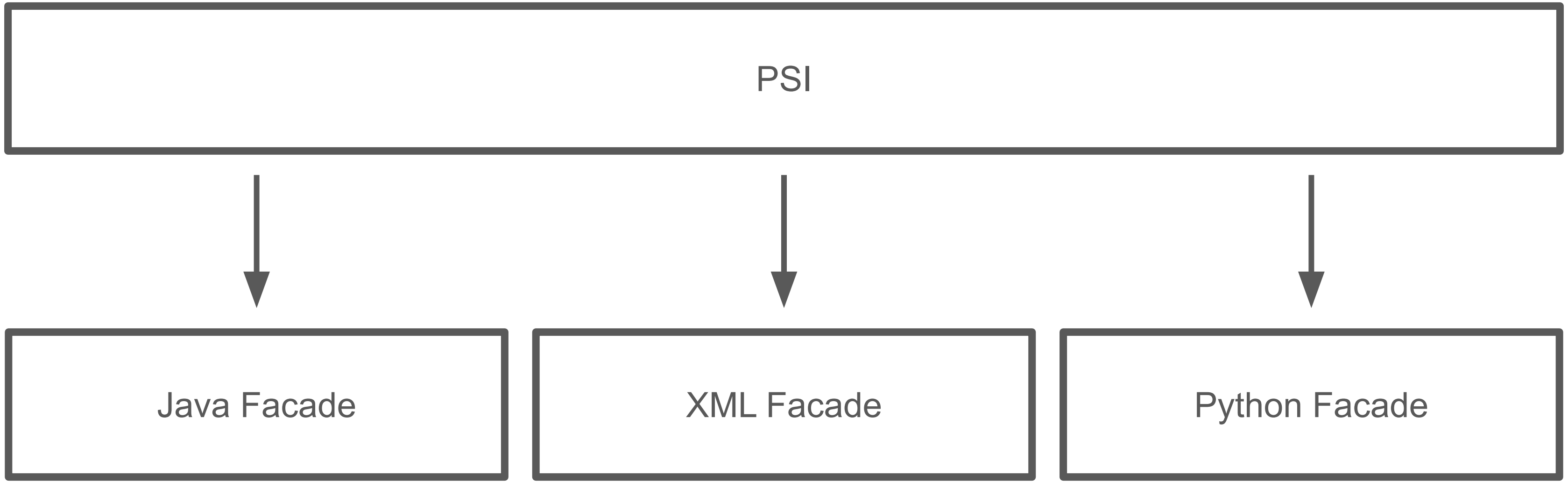


```
fun getProbeStatesForFile(file: String): List<ProbeState> {
    return temporaryState.lockedProbes.filter { it.file == file }
}
```



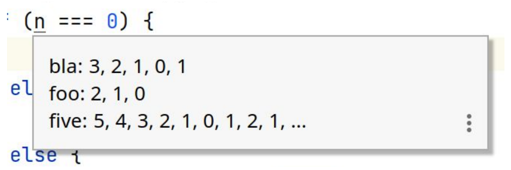
```
v RETURN
    PsiElement(return)
    PsiWhiteSpace
  v DOT_QUALIFIED_EXPRESSION
    v DOT_QUALIFIED_EXPRESSION
      > REFERENCE_EXPRESSION
        PsiElement(DOT)
      v REFERENCE_EXPRESSION
          PsiElement(IDENTIFIER)
      PsiElement(DOT)
    v CALL_EXPRESSION
      v REFERENCE_EXPRESSION
          PsiElement(IDENTIFIER)
```

Intellij Plugin



Intellij Plugin - Hover Probes

1. Connect to LSP when example activated
2. Find all possible probes (through PSI)
3. Send possible probes to graalvm
4. Store values
5. Send again on save
6. On hover: show values

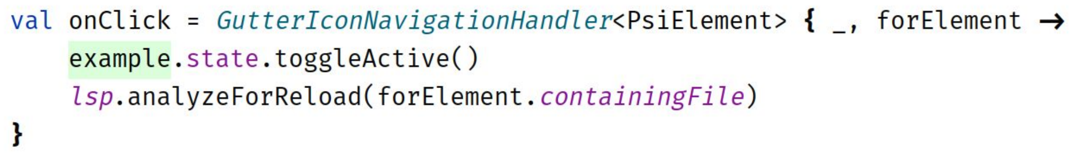


```
(n === 0) {
el
else {
```

```
bla: 3, 2, 1, 0, 1
foo: 2, 1, 0
five: 5, 4, 3, 2, 1, 0, 1, 2, 1, ...
```

Intellij Plugin - Hover Probes

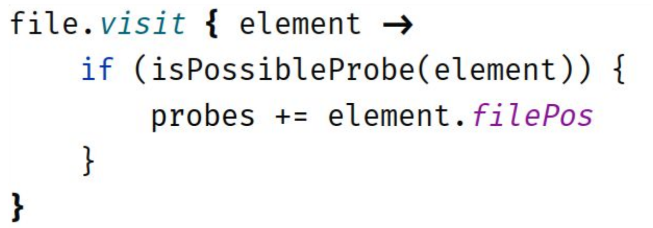
1. **Connect to LSP when example activated**
2. Find all possible probes (through PSI)
3. Send possible probes to graalvm
4. Store values
5. Send again on save
6. On hover: show values



```
val onClick = GutterIconNavigationHandler<PsiElement> { _, forElement →
    example.state.toggleActive()
    lsp.analyzeForReload(forElement.containingFile)
}
```

Intellij Plugin - Hover Probes

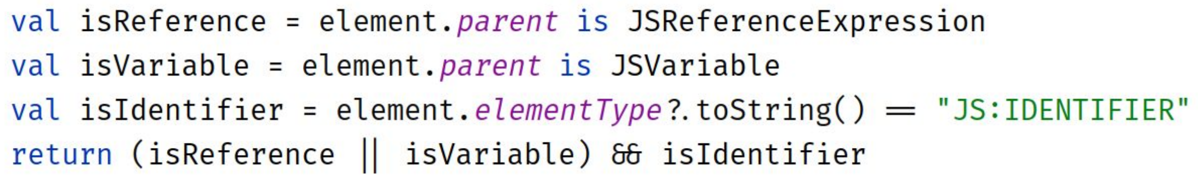
1. Connect to LSP when example activated
2. **Find all possible probes (through PSI)**
3. Send possible probes to graalvm
4. Store values
5. Send again on save
6. On hover: show values



```
file.visit { element →
    if (isPossibleProbe(element)) {
        probes += element.filePos
    }
}
```

Intellij Plugin - Hover Probes

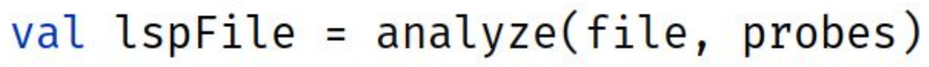
1. Connect to LSP when example activated
2. **Find all possible probes (through PSI)**
3. Send possible probes to graalvm
4. Store values
5. Send again on save
6. On hover: show values



```kotlin
val isReference = element.parent is JSReferenceExpression
val isVariable = element.parent is JSVariable
val isIdentifier = element.elementType?.toString() == "JS:IDENTIFIER"
return (isReference || isVariable) && isIdentifier
```

Intellij Plugin - Hover Probes

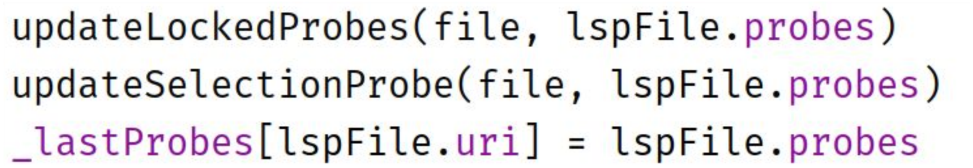
1. Connect to LSP when example activated
2. Find all possible probes (through PSI)
3. **Send possible probes to graalvm**
4. Store values
5. Send again on save
6. On hover: show values



```
val lspFile = analyze(file, probes)
```

Intellij Plugin - Hover Probes

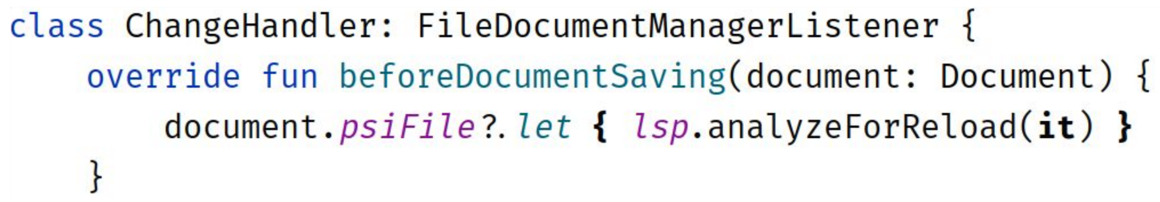
1. Connect to LSP when example activated
2. Find all possible probes (through PSI)
3. Send possible probes to graalvm
4. **Store values**
5. Send again on save
6. On hover: show values



```
updateLockedProbes(file, lspFile.probes)
updateSelectionProbe(file, lspFile.probes)
_lastProbes[lspFile.uri] = lspFile.probes
```

Intellij Plugin - Hover Probes

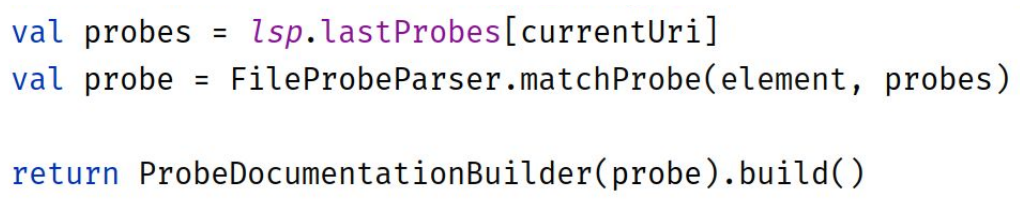
1. Connect to LSP when example activated
2. Find all possible probes (through PSI)
3. Send possible probes to graalvm
4. Store values
5. **Send again on save**
6. On hover: show values



```kotlin
class ChangeHandler: FileDocumentManagerListener {
    override fun beforeDocumentSaving(document: Document) {
        document.psiFile?.let { lsp.analyzeForReload(it) }
    }
```

Intellij Plugin - Hover Probes

1. Connect to LSP when example activated
2. Find all possible probes (through PSI)
3. Send possible probes to graalvm
4. Store values
5. Send again on save
6. **On hover: show values**



```kotlin
val probes = lsp.lastProbes[currentUri]
val probe = FileProbeParser.matchProbe(element, probes)

return ProbeDocumentationBuilder(probe).build()
```

Future Work

“Make Babylonian Programming available for everyone”

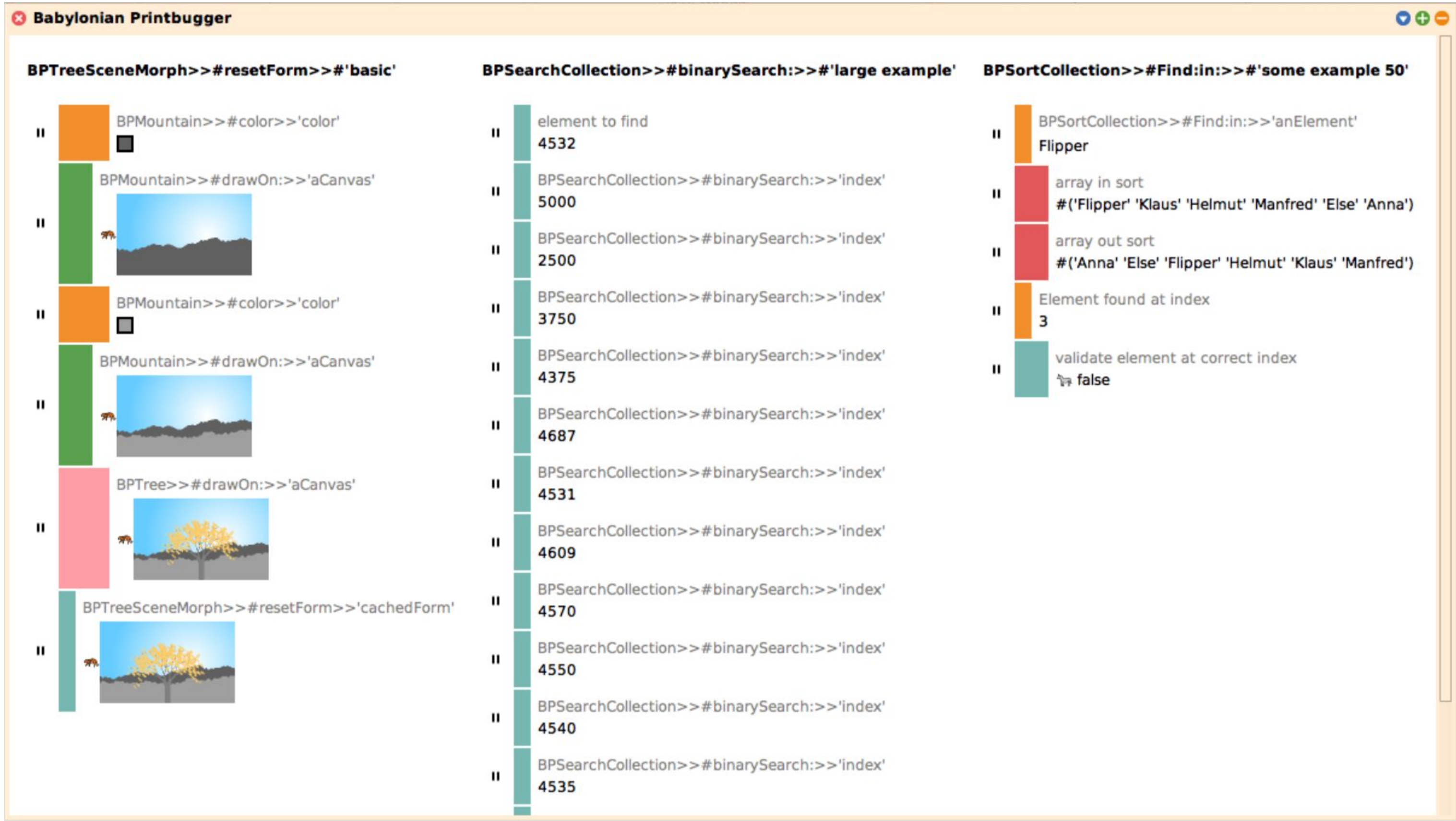
Limitations

- Runtime in GraalVM is not preserved $\longrightarrow$ No Live Specimen
- Communication between GraalVM and Intellij requires serialization
- Different environment $\longrightarrow$ Build Systems and Package Managers don’t work
- Some GraalVM Features are experimental and unstable

Supported Languages

- Intellij supports most languages out of the box, can be extended with plugins
- Intellij Plugin and Graal-Extension can be language agnostic
- GraalVM Language Server supports (out of box) Javascript, Ruby, Python

Additional Features



Babylonian Printbugger

BPTreeSceneMorph>>#resetForm>>#'basic'

BPMountain>>#color>>'color'

BPMountain>>#drawOn:>>'aCanvas'

BPMountain>>#color>>'color'

BPMountain>>#drawOn:>>'aCanvas'

BPTree>>#drawOn:>>'aCanvas'

BPTreeSceneMorph>>#resetForm>>'cachedForm'

BPSearchCollection>>#binarySearch:>>#'large example'

element to find
4532

BPSearchCollection>>#binarySearch:>>'index'
5000

BPSearchCollection>>#binarySearch:>>'index'
2500

BPSearchCollection>>#binarySearch:>>'index'
3750

BPSearchCollection>>#binarySearch:>>'index'
4375

BPSearchCollection>>#binarySearch:>>'index'
4687

BPSearchCollection>>#binarySearch:>>'index'
4531

BPSearchCollection>>#binarySearch:>>'index'
4609

BPSearchCollection>>#binarySearch:>>'index'
4570

BPSearchCollection>>#binarySearch:>>'index'
4550

BPSearchCollection>>#binarySearch:>>'index'
4540

BPSearchCollection>>#binarySearch:>>'index'
4535

BPSortCollection>>#Find:in:>>#'some example 50'

BPSortCollection>>#Find:in:>>'anElement'
Flipper

array in sort
#('Flipper' 'Klaus' 'Helmut' 'Manfred' 'Else' 'Anna')

array out sort
#('Anna' 'Else' 'Flipper' 'Helmut' 'Klaus' 'Manfred')

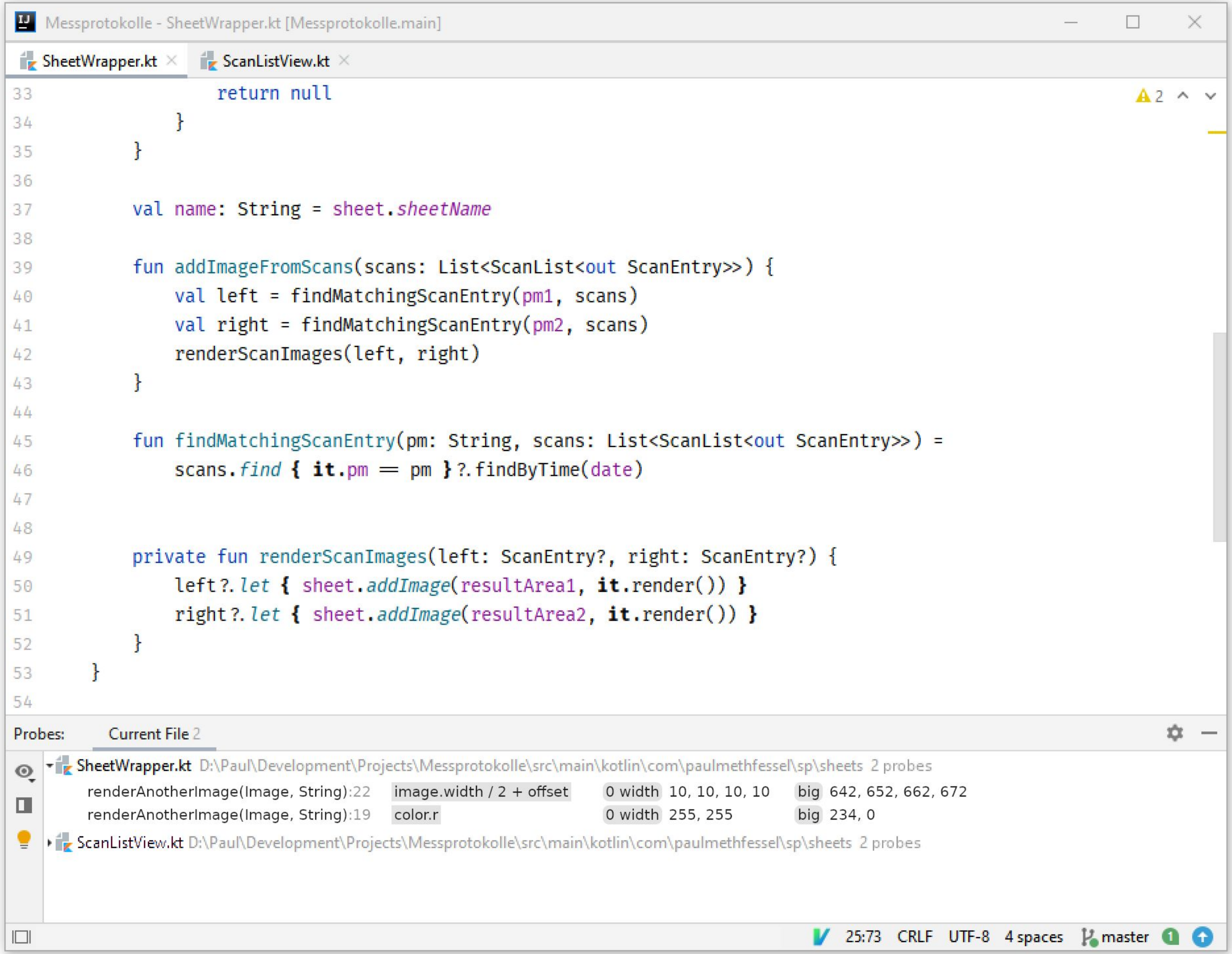
Element found at index
3

validate element at correct index
false

"Live Printf" by Joana Bergsiek and Lina Urban

HPI

Additional Features



Messprotokolle - SheetWrapper.kt [Messprotokolle.main]

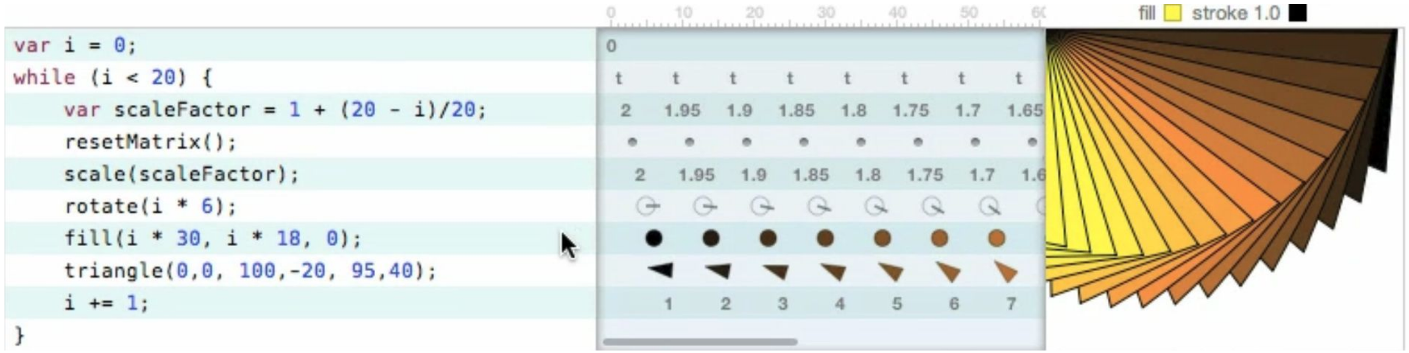
SheetWrapper.kt ScanListView.kt

```
                return null
            }
        }

        val name: String = sheet.sheetName

        fun addImageFromScans(scans: List<ScanList<out ScanEntry>>) {
            val left = findMatchingScanEntry(pm1, scans)
            val right = findMatchingScanEntry(pm2, scans)
            renderScanImages(left, right)
        }

        fun findMatchingScanEntry(pm: String, scans: List<ScanList<out ScanEntry>>) =
            scans.find { it.pm == pm }?.findByTime(date)


        private fun renderScanImages(left: ScanEntry?, right: ScanEntry?) {
            left?.let { sheet.addImage(resultArea1, it.render()) }
            right?.let { sheet.addImage(resultArea2, it.render()) }
        }
    }

```

```
Probes:   Current File 2

SheetWrapper.kt D:\Paul\Development\Projects\Messprotokolle\src\main\kotlin\com\paulmethfessel\sp\sheets 2 probes
    renderAnotherImage(Image, String):22   image.width / 2 + offset   0 width  10, 10, 10, 10   big  642, 652, 662, 672
    renderAnotherImage(Image, String):19   color.r                    0 width  255, 255         big  234, 0
ScanListView.kt D:\Paul\Development\Projects\Messprotokolle\src\main\kotlin\com\paulmethfessel\sp\sheets 2 probes
```

25:73 CRLF UTF-8 4 spaces master

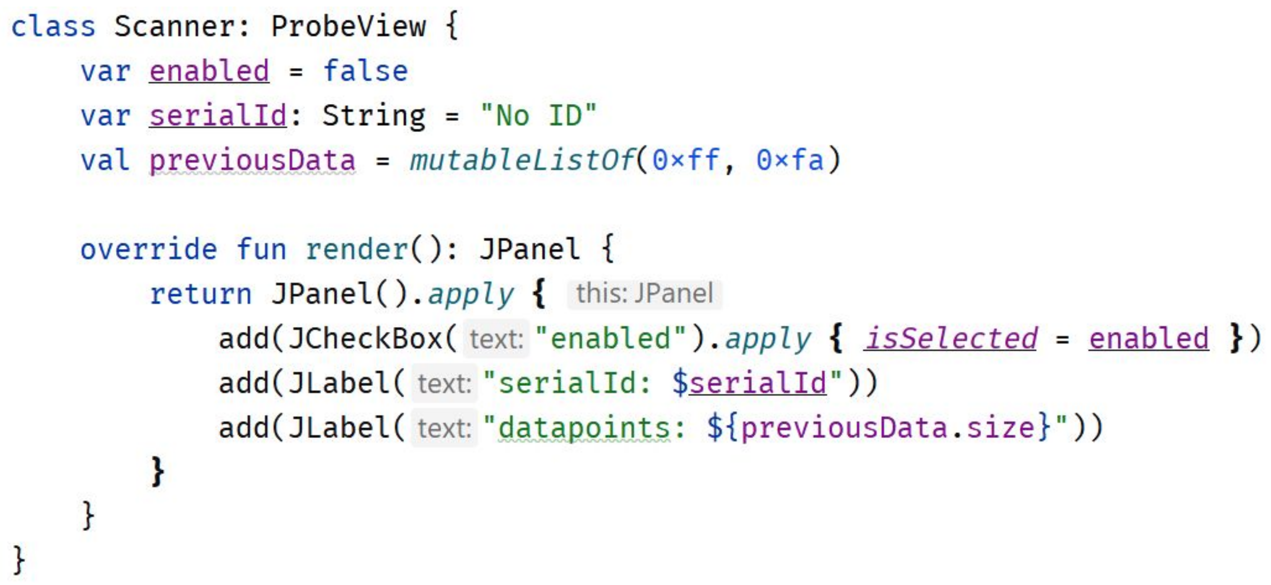
Additional Features



```
var i = 0;
while (i < 20) {
    var scaleFactor = 1 + (20 - i)/20;
    resetMatrix();
    scale(scaleFactor);
    rotate(i * 6);
    fill(i * 30, i * 18, 0);
    triangle(0,0, 100,-20, 95,40);
    i += 1;
}
```

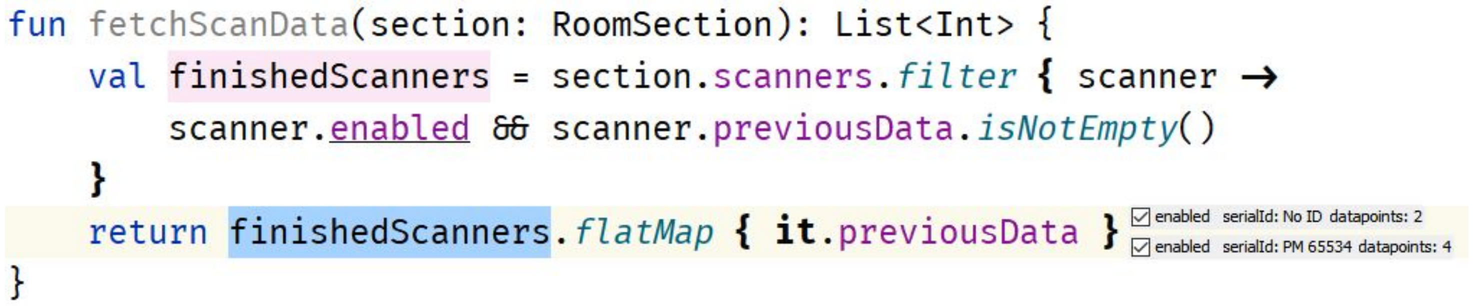
```
0
t     t     t     t     t     t     t     t
2     1.95  1.9   1.85  1.8   1.75  1.7   1.65
2     1.95  1.9   1.85  1.8   1.75  1.7   1.6
      1     2     3     4     5     6     7
```

Additional Features



```
class Scanner: ProbeView {
    var enabled = false
    var serialId: String = "No ID"
    val previousData = mutableListOf(0xff, 0xfa)

    override fun render(): JPanel {
        return JPanel().apply {  this: JPanel
            add(JCheckBox( text: "enabled").apply { isSelected = enabled })
            add(JLabel( text: "serialId: $serialId"))
            add(JLabel( text: "datapoints: ${previousData.size}"))
        }
    }
}
```

Additional Features



```kotlin
fun fetchScanData(section: RoomSection): List<Int> {
    val finishedScanners = section.scanners.filter { scanner →
        scanner.enabled && scanner.previousData.isNotEmpty()
    }
    return finishedScanners.flatMap { it.previousData } ☑ enabled  serialId: No ID  datapoints: 2
                                                        ☑ enabled  serialId: PM 65534  datapoints: 4
}
```

Acknowledgements

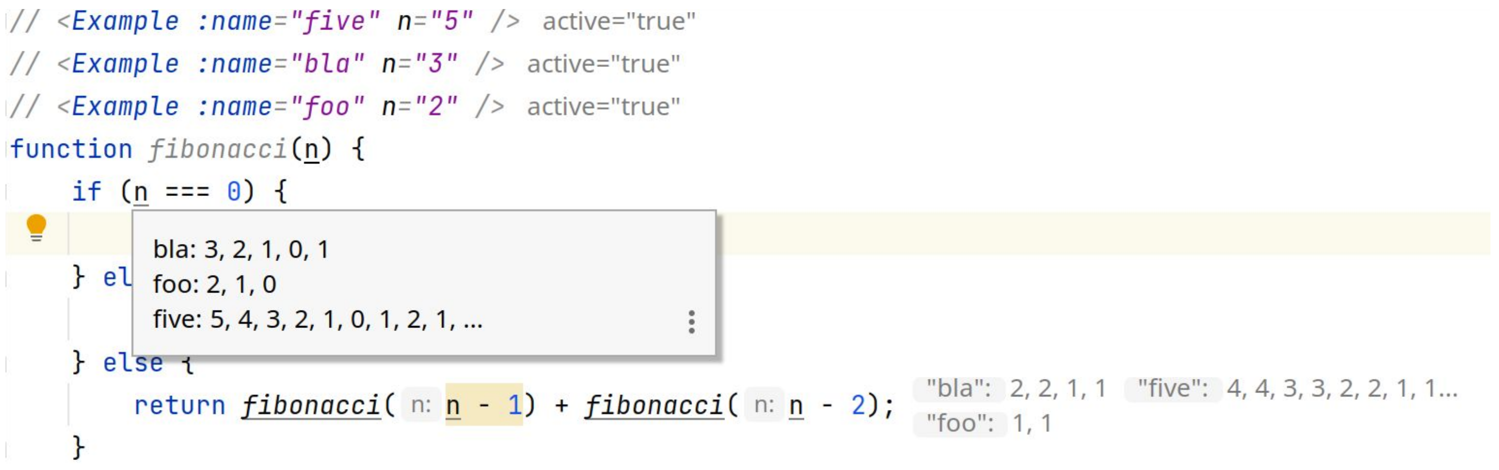
Patrick Rein

for advice and help on the project

Fabio Niephaus

for helping with questions regarding GraalVM and the Polyglot extension

Thanks for listening, any questions?



```
// <Example :name="five" n="5" />  active="true"
// <Example :name="bla" n="3" />  active="true"
// <Example :name="foo" n="2" />  active="true"
function fibonacci(n) {
    if (n === 0) {

    } el

    } else {
        return fibonacci( n: n - 1) + fibonacci( n: n - 2);   "bla": 2, 2, 1, 1   "five": 4, 4, 3, 3, 2, 2, 1, 1...
                                                              "foo": 1, 1
    }
```

```
bla: 3, 2, 1, 0, 1
foo: 2, 1, 0
five: 5, 4, 3, 2, 1, 0, 1, 2, 1, ...
```