Zeppelin Helium : Spell

1ambda

ZEPL

March 2, 2017

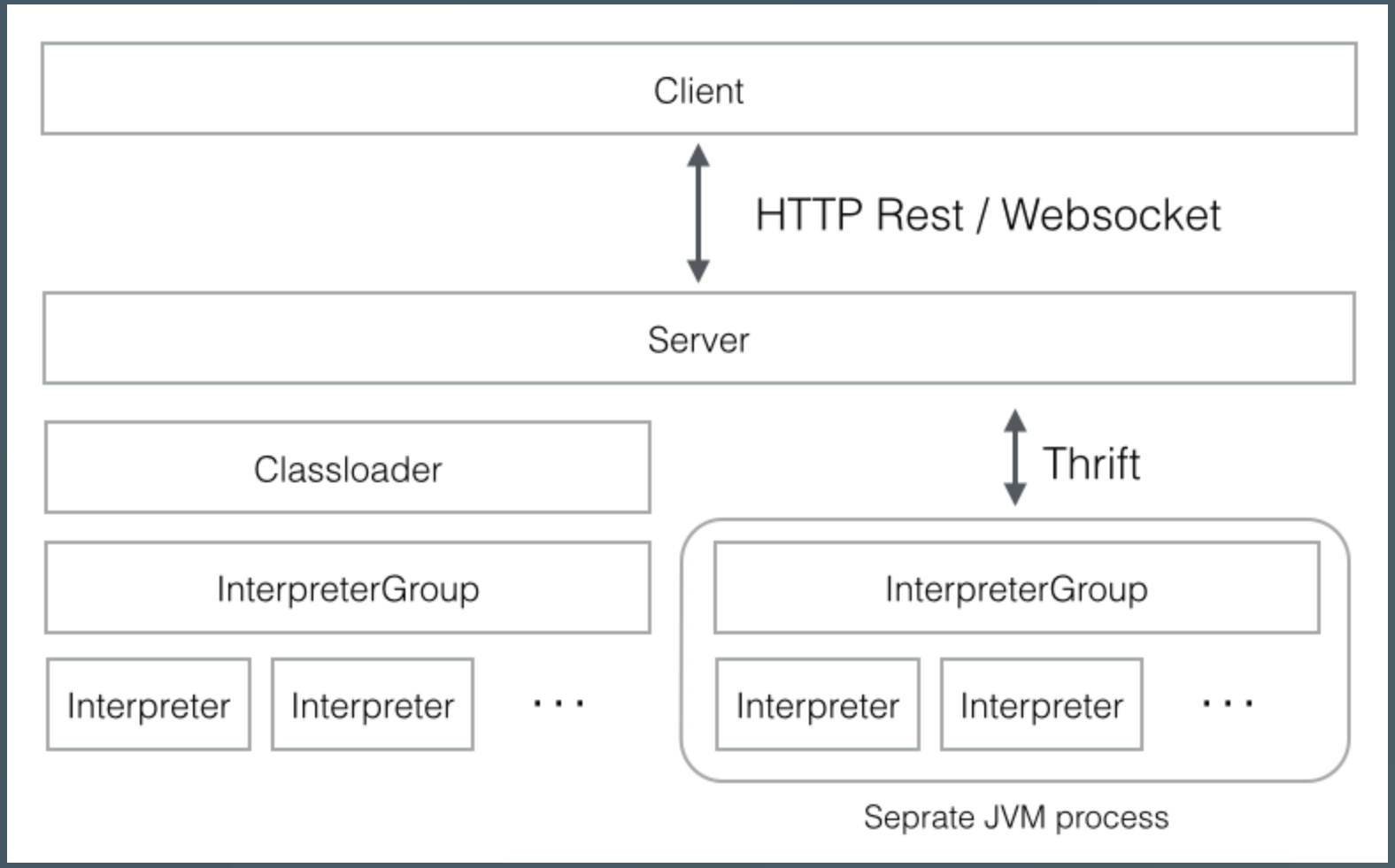
What is Spell?

Frontend interpreter runs on browser not in backend.

- **pluggable**
- **written in javascript**
- **can be display system**
- available in **0.8.0-SNAPSHOT** (ZEPPELIN #1940)

What?! 

Background: Interpreter



Background: Interpreter (cont.)

- Interpreter consumes paragraph text and render output.
- Each interpreter has magic. For example, the markdown interpreter uses `%markdown`

```
%spark

println("Hello, Zeppelin!")
```

```
%markdown

## Hello, Zeppelin
```

Background: Display System

- Display system renders result which already translated by backend interpreters **on browser** (Basic Display System: `%html`, `%table`, `%angular`)

```
%spark

val total = 195
println(s"%html <h3>result is ${total}</h3>")

// will be interpreted by spark interpreter
// `%html <h3>total value is 195<h3>`
```

- Can we do better? (e.g `%markdown` display type)

Motivation: Spell

- How can i pass variables from spark to markdown in Zeppelin
- Do we really need backend markdown interpreter? (awesome js markdown library)
- What if we implement `%markdown` as a **frontend** interpreter?
- Then, **we can also use them as display system** because it runs on browser like `%html` 😏

This is Spell 😎

Frontend interpreter runs on browser not in backend.

- **pluggable**: can be installed / removed easily (Helium Online Registry)
- **written in javascript**: can utilise exisiting libraries (flowchart, sigmajs, vega, papaparse, ...)
- **can be display system as well** (ZEPPELIN-2089)

DEMO

1. zeppelin-flowchart-spell
2. zeppelin-csv-spell
3. zeppelin-json-spell
4. zeppelin-translator-spell

I wanna create my own spell 😯

- Doc: How to create new spell
- Basic Examples
- Published Spells
- Configurtion Support: #1982
- Ideas: `%slack`, `%sigma`, `%vega`

DEMO

1. zeppelin-echo-spell
2. zeppelin-markdown-spell

Creating Spell: interpret()

- every spell takes paragraph text (string)
- returns SpellResult
- spell-base.js
- example: zeppelin-echo-spell

```
interpret(paragraphText) {
  return new SpellResult(paragraphText)
}
```

Creating Spell: SpellResult

- SpellResult(data, displayType)
- displayType can be optional (default is `%text`)

```
import {
    SpellBase,
    SpellResult,
    DefaultDisplayType,
} from 'zeppelin-spell';

interpret(paragraphText) {
  const html = `<h2>${paragraphText}</h2>`
  return new SpellResult(html, DefaultDisplayType.HTML)
}
```

Creating Spell: SpellResult (cont.)

- SpellResult supports multiple results: add()
- See also: Multiple paragraph results (PR #1658)

```
interpret(paragraphText) {
    const html = '<h4>Hello</h4>'
    const text = 'Spell'

    return new SpellResult()
     .add(html, DefaultDisplayType.HTML)
     .add(text, DefaultDisplayType.TEXT /** optional */)
}
```

Creating Spell: SpellResult (cont.)

- pass Promise for **API call** (async)
- psss Function which takes elem id to **draw DOM**
- default display types

data	displayType	example
String	ALL (except `%element`)	markdown-spell
Promise	ALL (except `%element`)	translator-spell
Function	ELEMENT (`%element`)	flowchart-spell

Creating Spell: Configuration

- Actually, interpret() takes config as the second argument

```
interpret(paragraphText, config) { ... }
```

- Define config specification in package.json like

```
  "config": {
    "repeat": {
      "type": "number",
      "description": "How many times to repeat",
      "defaultValue": 1
    }
  },
```

(unresolved) Helium, Spell Issues

we need your help 😭

- ZEPPELIN-2089: Use spell as display system with backend interpreters
- ZEPPELIN-2088: Evaluate helium bundles one by one
- ZEPPELIN-2122: Add execution time to paragraphs executed by spell
- Other Issues

References

- Image: Zeppelin Interpreter Architecture

Resources

- Slide: github.com/1ambda/talk/2017/fly-zeppelin

Thanks 😀