



# The Future of IDEs

Tracy Miranda  
Kichwa Coders  
@tracymiranda

# Integrated Developer Environment

Build  
software  
stuff

Quickly,  
Scale well

Troubleshoot

# Integrated Developer Environment

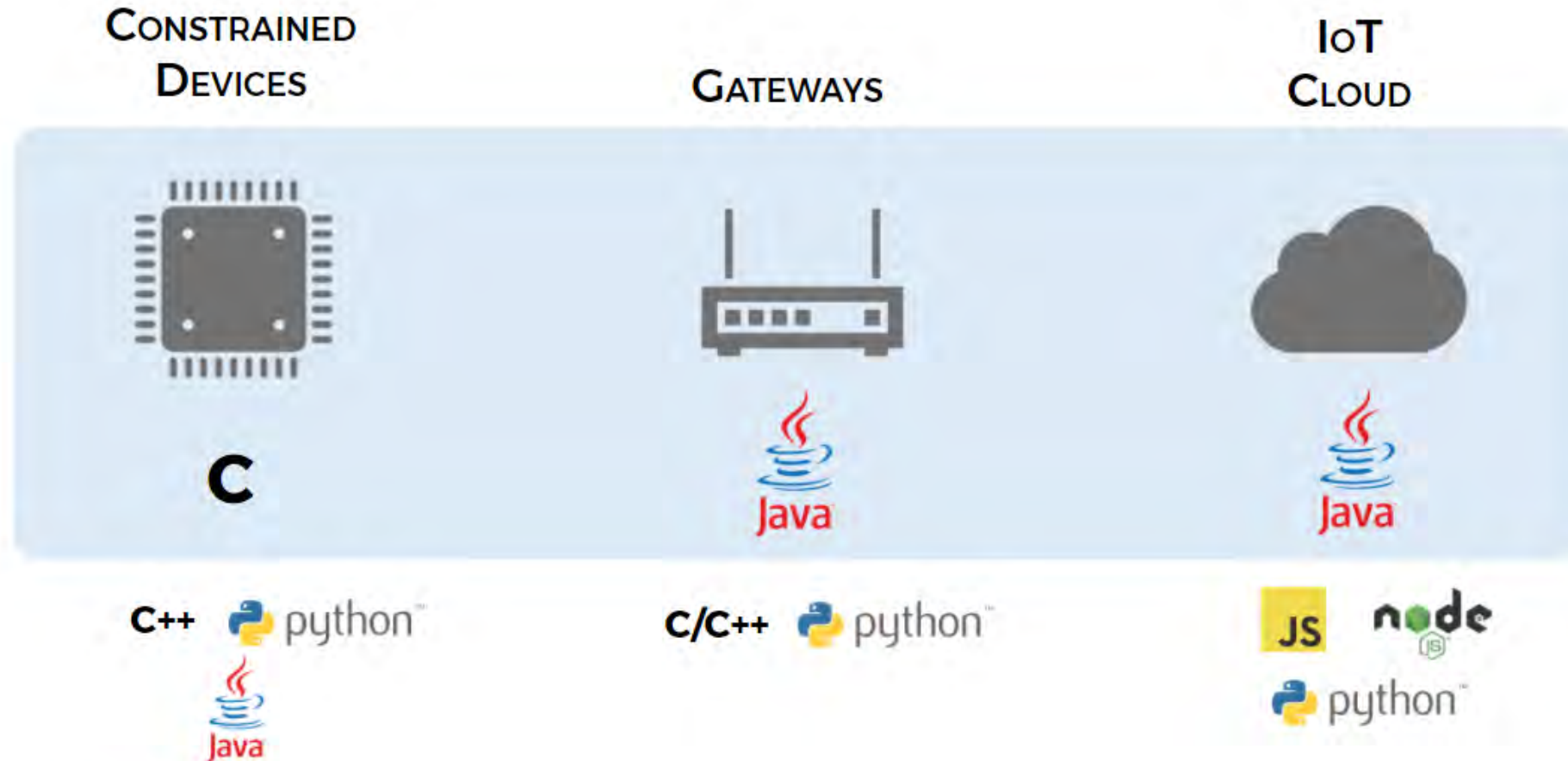


vs



**What will we be  
using to develop code  
in the next  
10-15 years?**

# What Language for IoT ?



C



python™

Assembler

C#



Scala



php

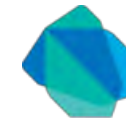


Swift

Verilog



BASH  
THE BOURNE-AGAIN SHELL



Dart



ERLANG

# Eclipse IDE

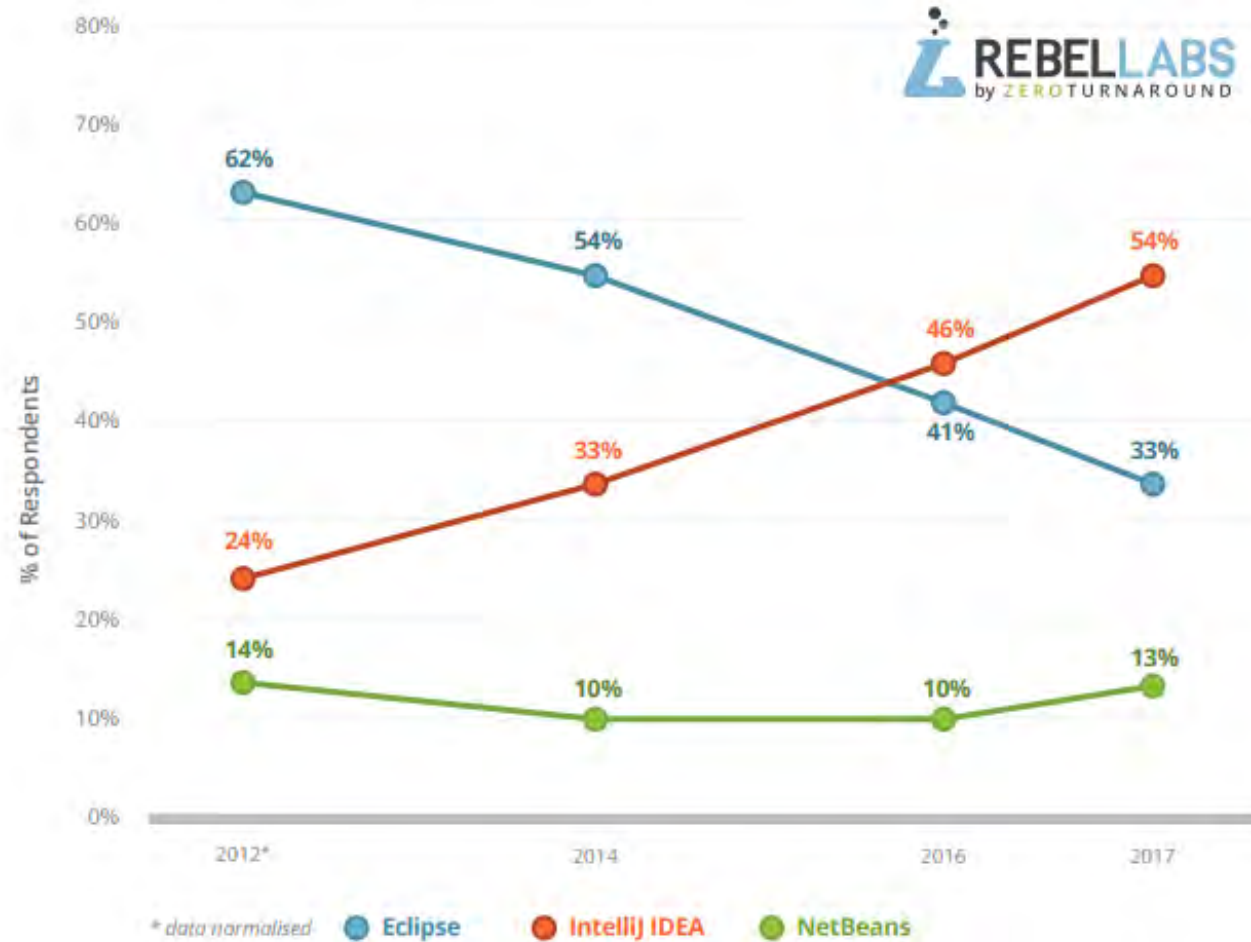
@eclipsejavaide



# To IDE or not to IDE?

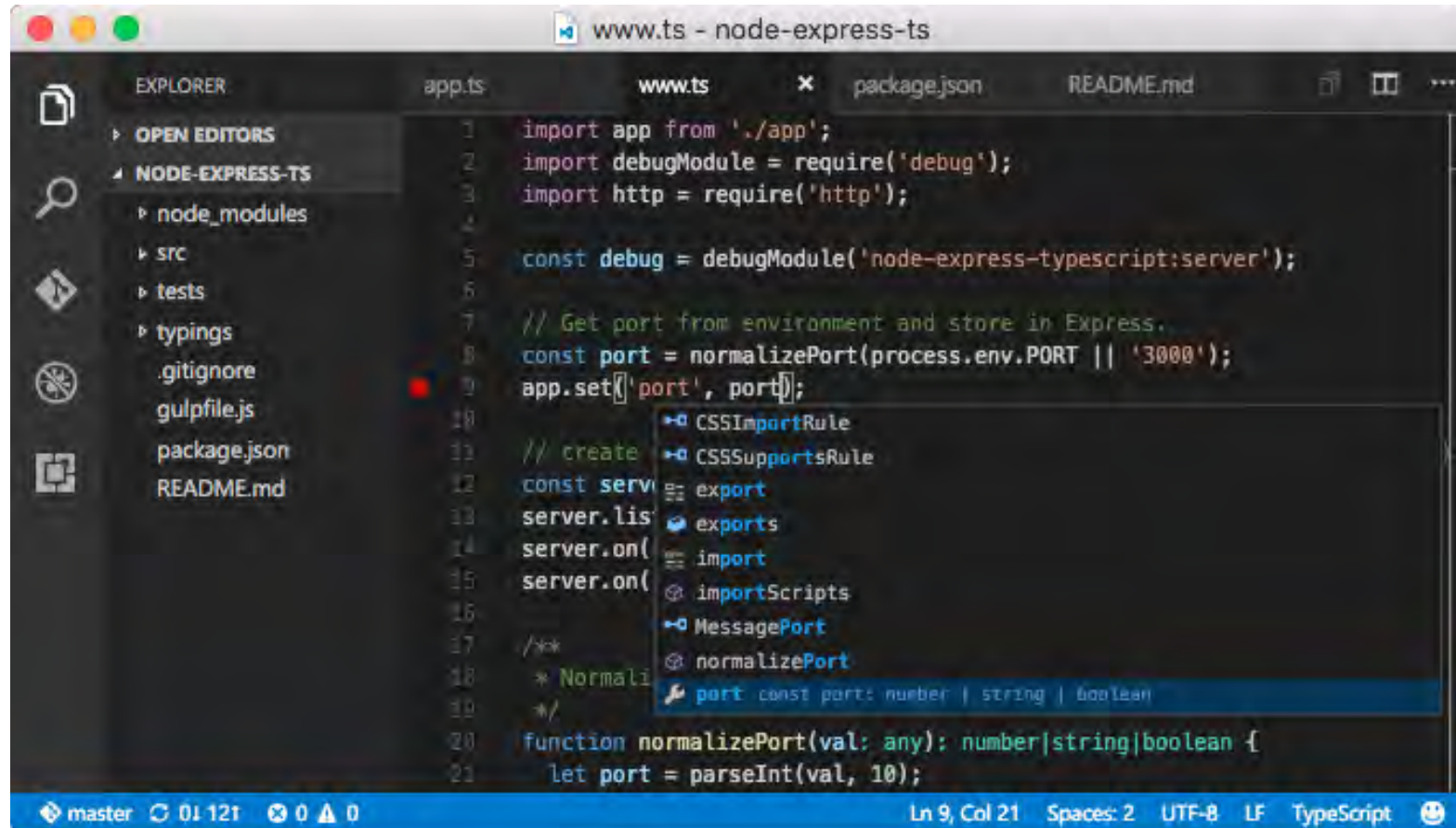


## Comparing 2012, 2014, 2016 and 2017 results: IDEs



<https://content.zereturnaround.com/rebellabs-reports/rebel-labs-developer-productivity-report-2017-2>

# VS Code



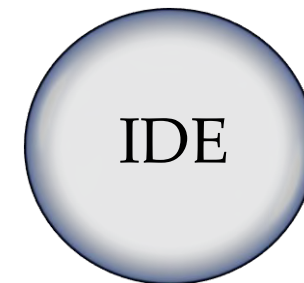
The screenshot shows the Visual Studio Code editor interface. The Explorer sidebar on the left displays the project structure for 'www.ts - node-express-ts', including folders like 'node\_modules', 'src', 'tests', and 'typings', and files like 'package.json' and 'README.md'. The main editor area shows the 'app.ts' file with the following code:

```
1 import app from './app';
2 import debugModule = require('debug');
3 import http = require('http');
4
5 const debug = debugModule('node-express-typescript:server');
6
7 // Get port from environment and store in Express.
8 const port = normalizePort(process.env.PORT || '3000');
9 app.set('port', port);
10
11 // create
12 const server = app.listen(port);
13 server.listen(port);
14 server.on('error', (err) => {
15   if (err.syscall !== 'listen') {
16     throw err;
17   }
18   // Normalized port is a string
19   let port = port;
20   function normalizePort(val: any): number|string|boolean {
21     let port = parseInt(val, 10);
```

A hover tooltip is visible over the variable 'port' on line 9, displaying a list of possible types: CSSImportRule, CSSSupportsRule, export, exports, import, importScripts, MessagePort, and normalizePort. The 'normalizePort' type is highlighted in blue.

The status bar at the bottom indicates the current file is 'app.ts', the cursor is at 'Ln 9, Col 21', and the file encoding is 'UTF-8 LF TypeScript'.

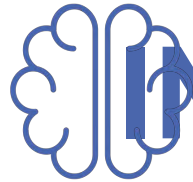
# The IDE Spectrum



# VISUAL (HIGHER LEVEL)



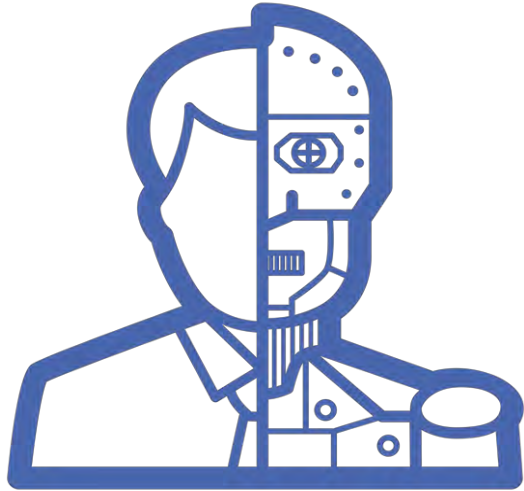
# ARTIFICIAL INTELLIGENCE



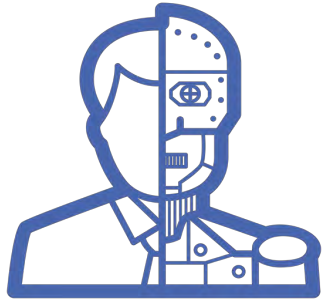
# CLOUD

# VISUAL (HIGHER LEVEL)

# Digital Twin



**Digital replica of physical assets, processes and systems. The digital representation provides both elements and dynamics of how an Internet of Things device operates and lives throughout its life cycle.**



**We need a higher level of abstraction for representing increasingly complex dynamic systems.**

# Node-Red?

The screenshot displays the Node-RED web interface. On the left, the 'input' palette includes nodes like inject, catch, status, link, mqtt, http, websocket, tcp, and udp. The 'output' palette includes debug, link, mqtt, and http response. The main workspace shows a flow named 'Flow 1' with the following components:

- ThingMonk** (Twitter node) connected to a **sentiment** node.
- everything is awesome** and **where is the coffee??** (inject nodes) also connected to the **sentiment** node.
- The **sentiment** node connects to a **switch** node.
- The **switch** node branches into two paths:
  - Path 1: **Count Positive** (function node) → **Positive total** (display node).
  - Path 2: **Count Negative** (function node) → **Negative Total** (display node).
- A **timestamp** node connects to a **Reset** function node.
- A **hello world** inject node connects to a **msg.payload** display node.

On the right, the 'debug' console shows the following log entries:

```
9/12/2017, 4:40:31 PM node: Negative Total
tweets/yoditstanton : msg.payload : number
37

9/12/2017, 4:42:26 PM node: Positive
tweets/BorisAdryan : msg.payload : string[137]

"The nice thing about #thingmonk being a
'friends and family' event are the frequent
back- and cross-references to previous related
talks."

9/12/2017, 4:42:26 PM node: Positive total
tweets/BorisAdryan : msg.payload : number
46

9/12/2017, 4:43:05 PM node: Positive
tweets/BrianLinuxing : msg.payload : string[138]

▶ "At #Thingmonk?@Like @Raspberry_Pi ?@Free
tickets to Saturday's Covent Garden PI Jam
Summer 17, 1 pm @dragonhall..
https://t.co/LC7NTdXUq2"

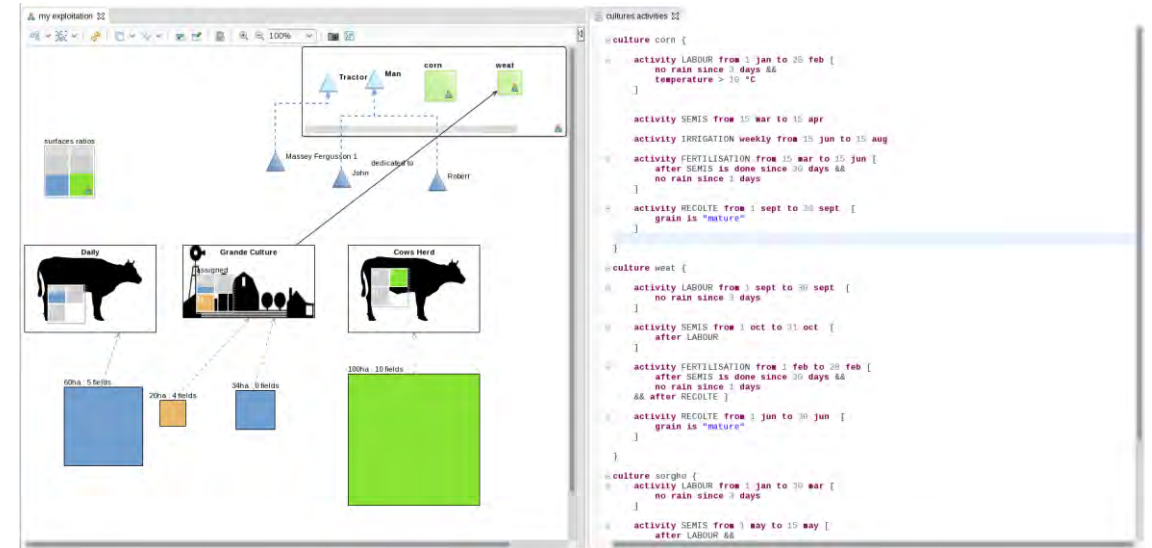
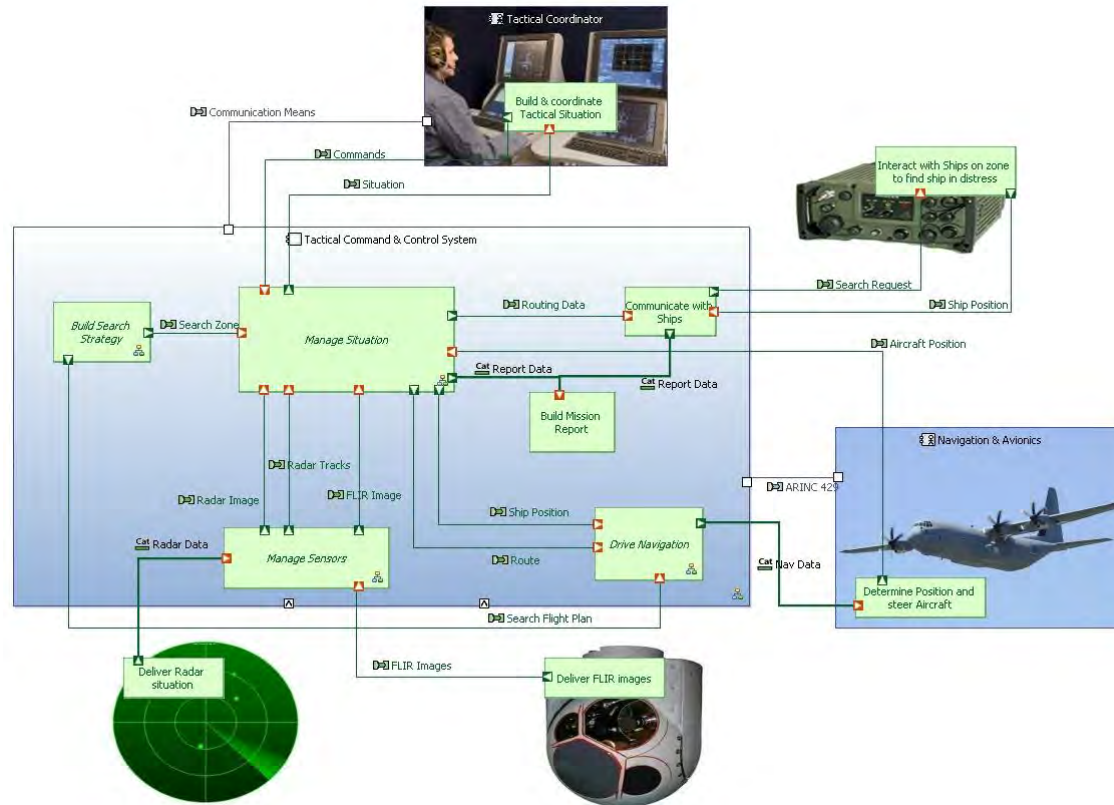
9/12/2017, 4:43:05 PM node: Positive total
tweets/BrianLinuxing : msg.payload : number
47

9/12/2017, 4:43:10 PM node: Negative
tweets/yoditstanton : msg.payload : string[69]

"'The community doesn't know what blockchain
is' @joepindar #thingmonk"
```



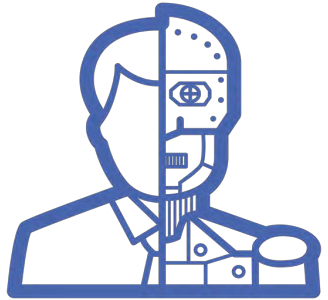
# Model Based Design



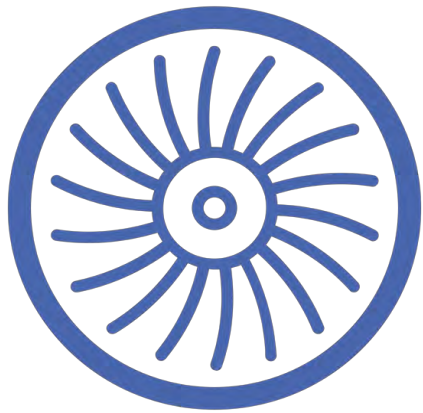
<https://www.eclipse.org/sirius/gallery.html>

# Visual Tools Should Be:

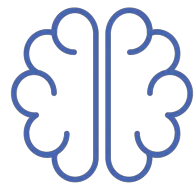
- Able to support hierarchy of abstraction layers
  - Allow users to work at optimal layer (e.g. drill down)
  - Transition to text layer
- Scriptable
- Debuggable



**We need a higher level of abstraction for representing increasingly complex dynamic systems.**



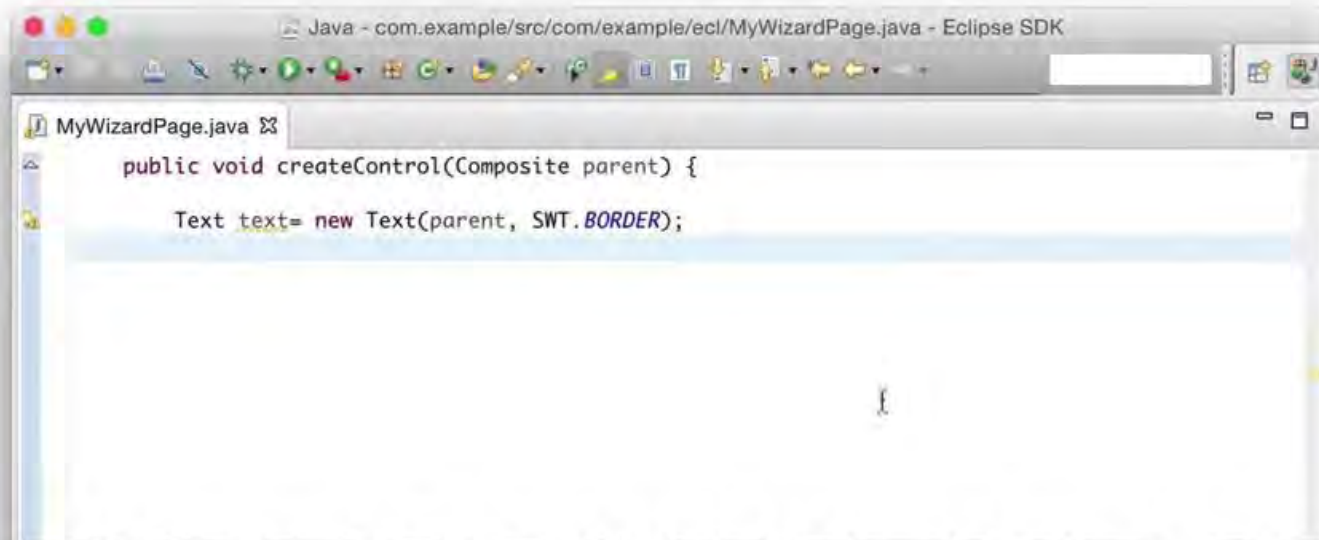
**Future IDEs need more  
visual programming  
models and less text**



# ARTIFICIAL INTELLIGENCE

# Focus on What Matters

The knowledge of thousands of devs in your code completion...



```
Java - com.example/src/com/example/ecl/MyWizardPage.java - Eclipse SDK  
MyWizardPage.java  
public void createControl(Composite parent) {  
    Text text= new Text(parent, SWT.BORDER);  
}
```

# CODE RECOMMENDATION

# Intelligent Completions: Code Recommenders



MyWizardPage.java

```
Button button = new Button(parent, SWT.PUSH);  
button.
```

- setText(String string) : void - Button 92%
- addSelectionListener(SelectionListener listener) : void - Button - 44%
- setLayoutData(Object layoutData) : void - Control - 39%

A large, curved green arrow points from the top right towards the "92%" value in the first suggestion.

# AI-Powered Bug Detection

A screenshot of a code editor window titled "MyWizardPage.java". The code shows a Java class definition: 

```
public class MyWizardPage extends WizardPage {  
  
    public MyWizardPage() {
```

 A red bug icon is positioned on the left side of the editor, pointing to the opening curly brace of the constructor. A red-bordered box contains the following text: "A super-call to DialogPage.setControl(Control) is missing. In 100% of direct subclasses of WizardPage, such a call was present." A green arrow points from the bottom left towards the bug icon, and a blue arrow points from the bottom left towards the red-bordered box.



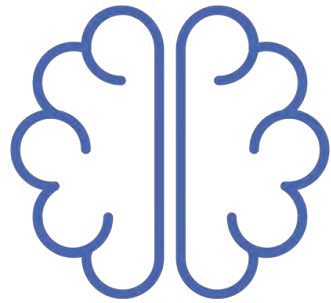
# Software Mining

Open source code  
mining

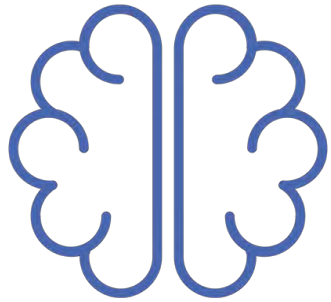
Configuration mining

Natural language  
sources





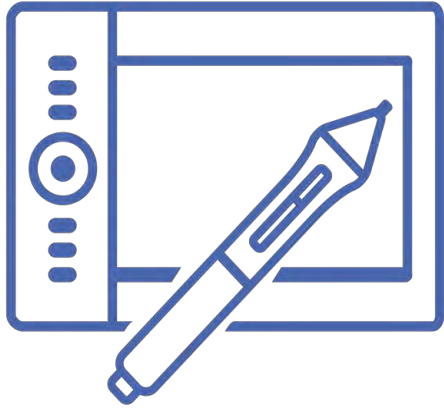
# Who wants AI powered tools?



**We have the technology**

**We have the data**

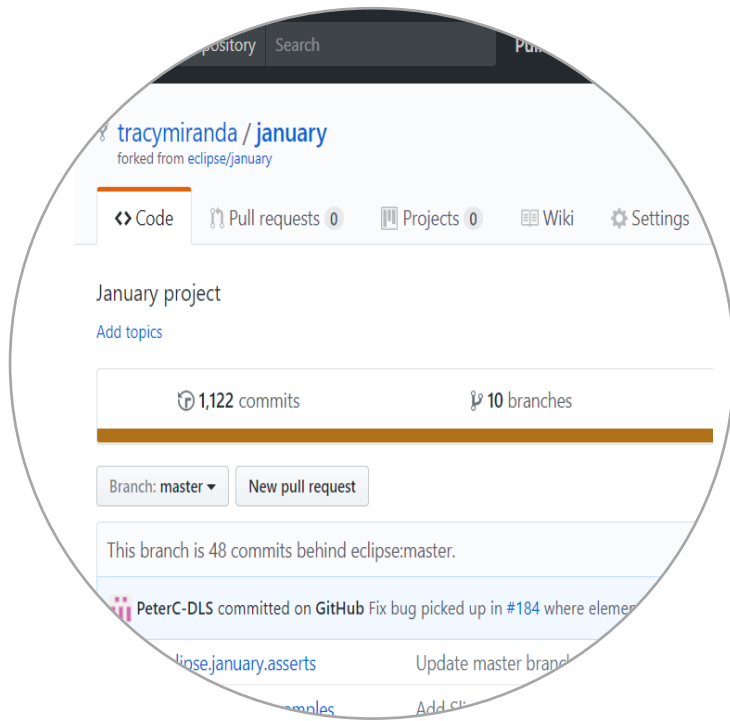
**We need the integration & design**



***“Design is not just what it looks like and feels like. Design is how it works” –Steve Jobs***

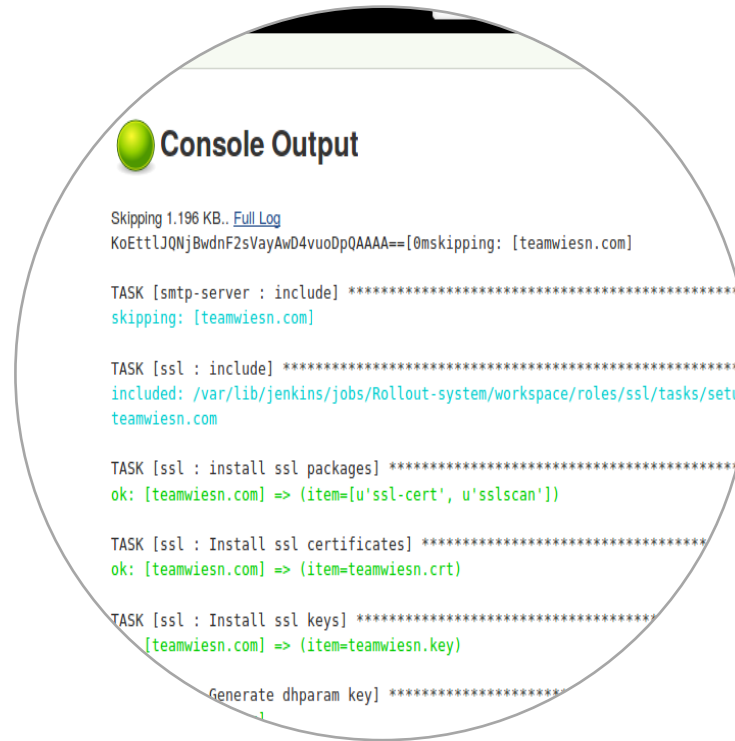


# DevOps In The Cloud



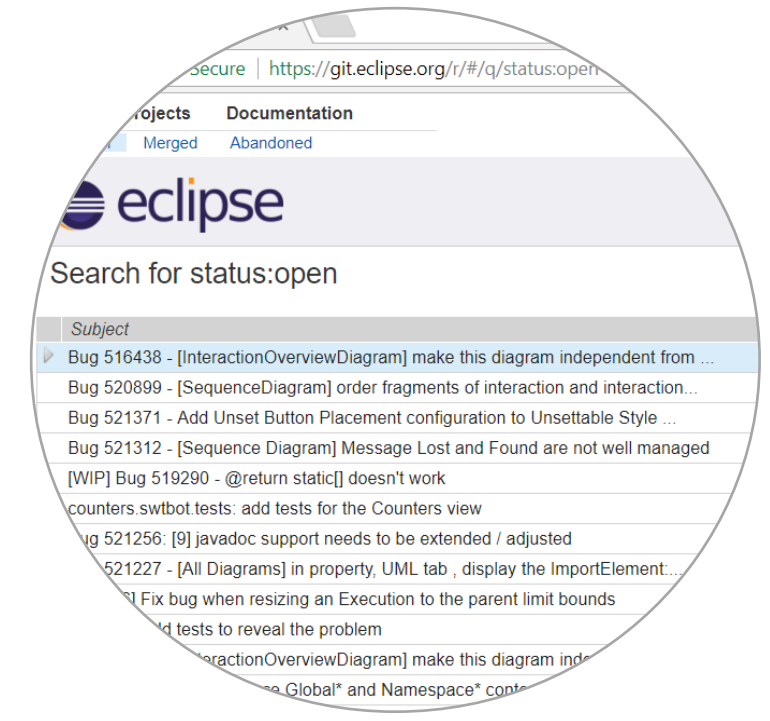
## Source Control

github, bitbucket, etc



## Builds & CI

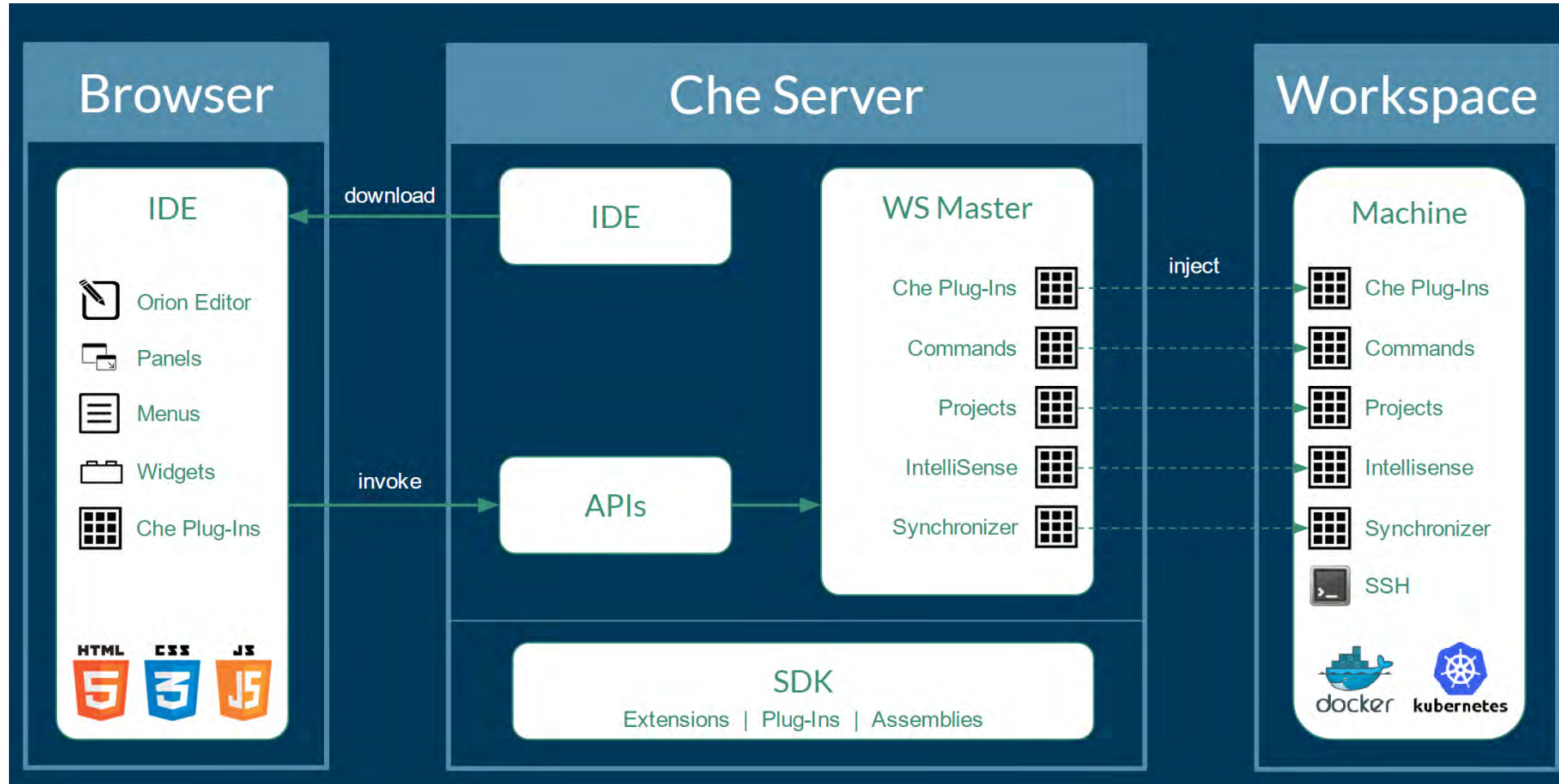
jenkins, hudson, travis, circleci etc



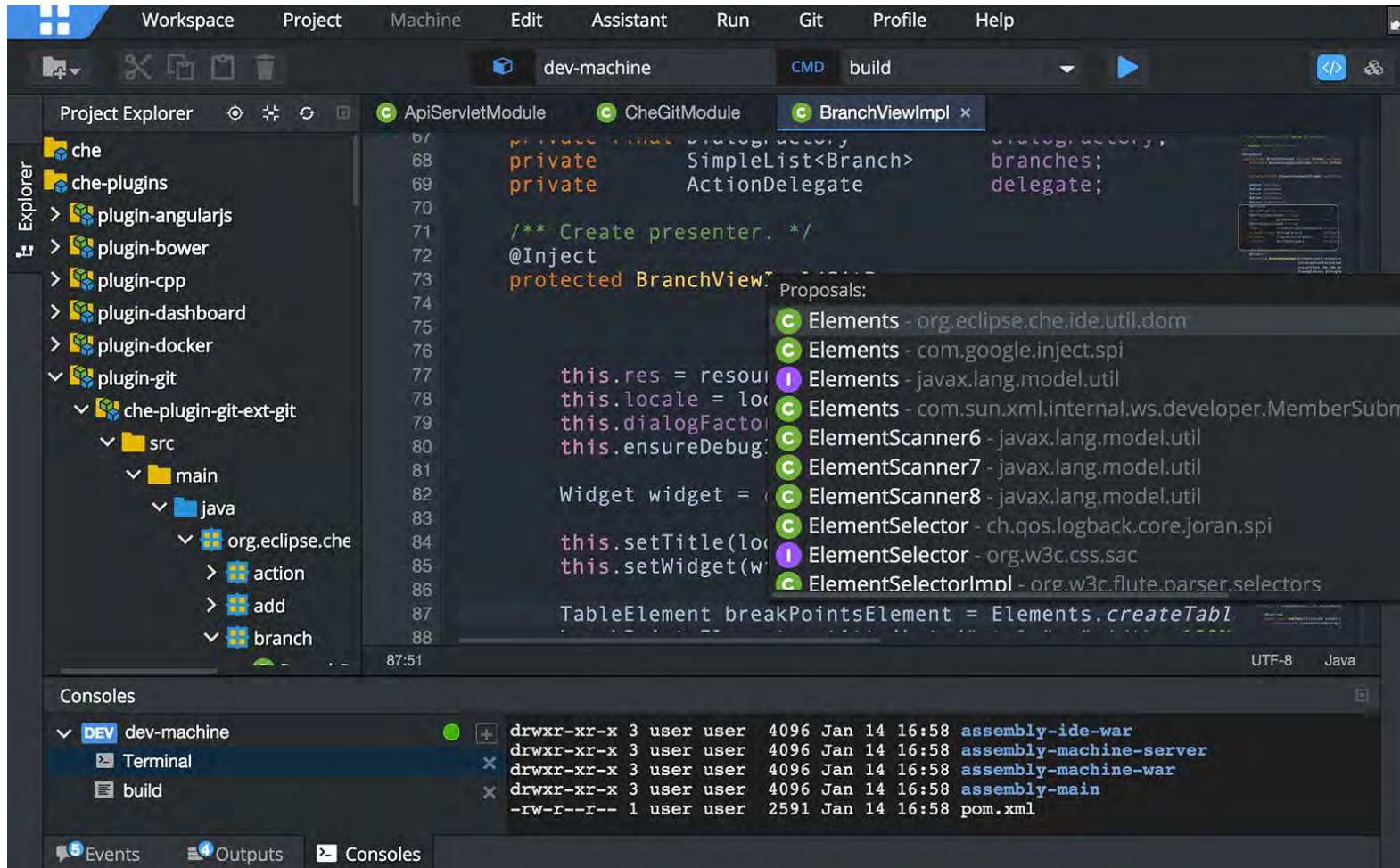
## Other

jira, bugzilla, gerrit etc

# Cloud Workspaces

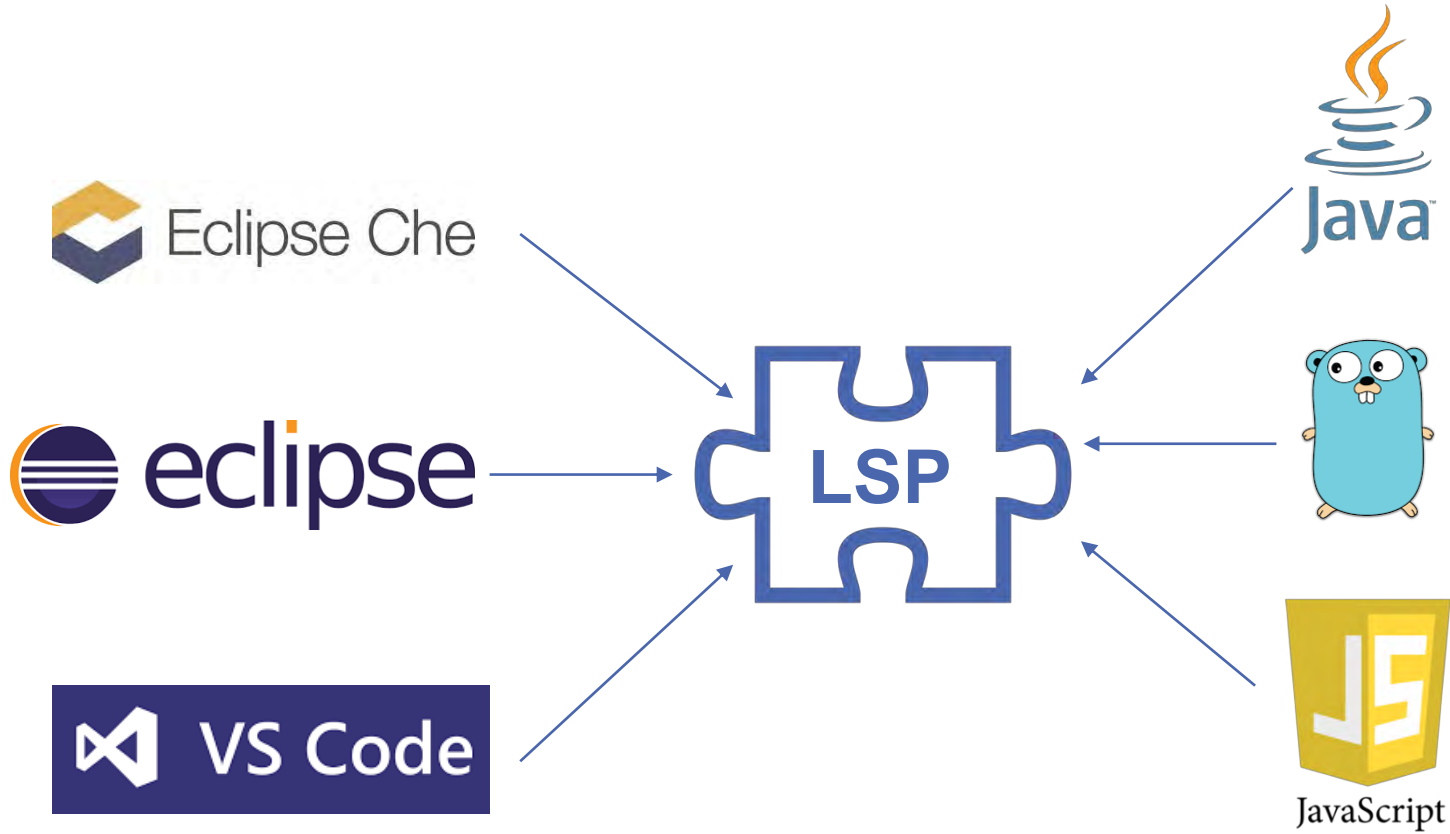


# Cloud Workspaces

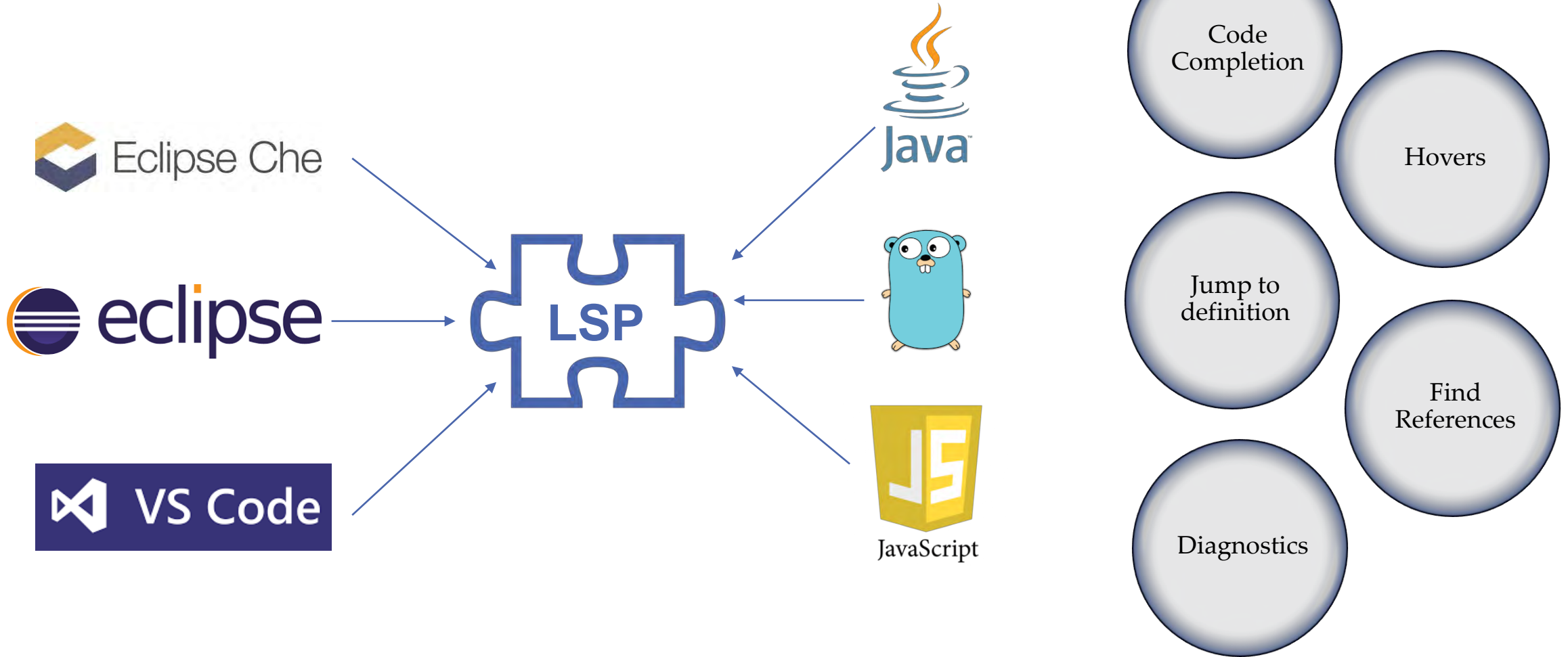




# Language Server Protocol

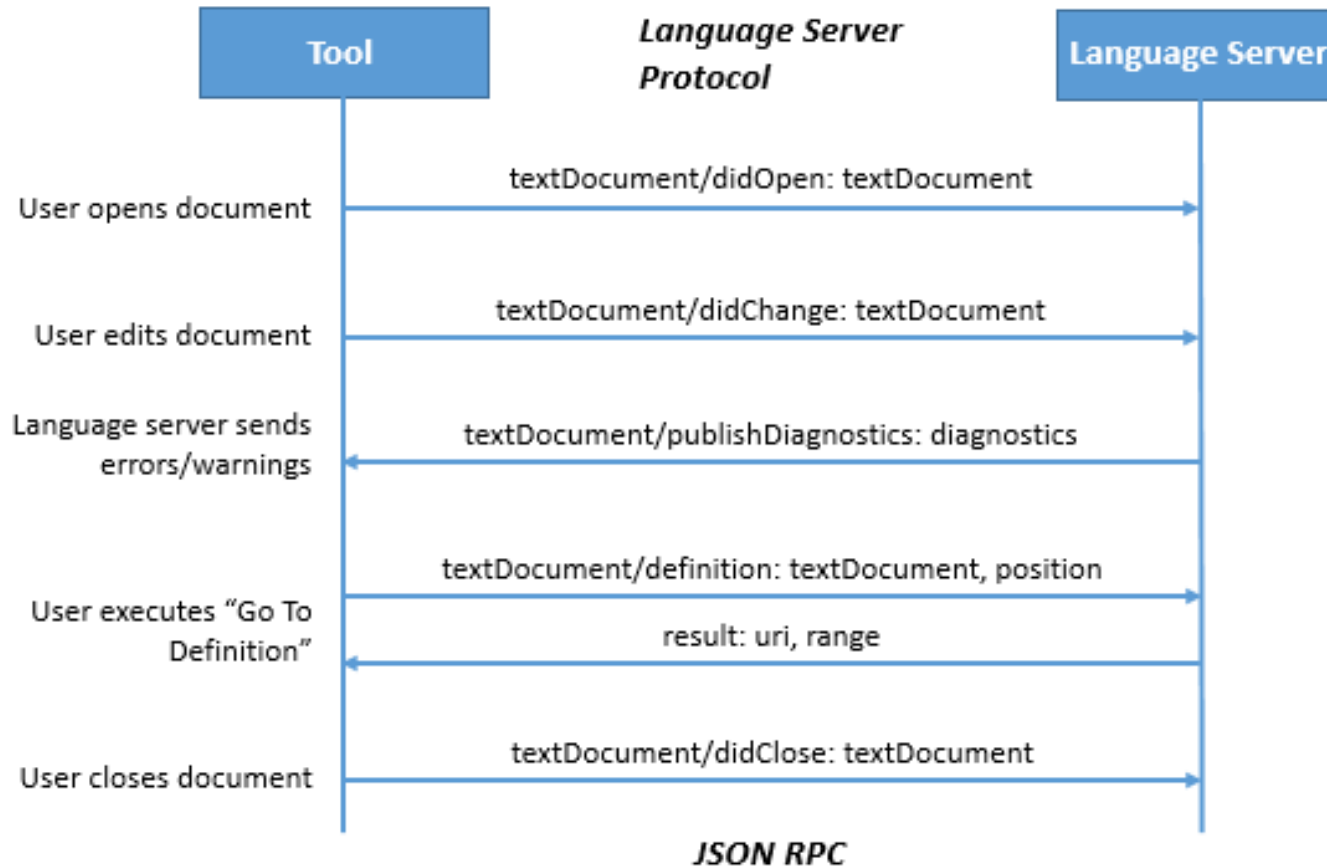


# Language Server Protocol



<https://github.com/Microsoft/language-server-protocol/wiki/Protocol-Implementations>

# Language Server Protocol



# Language Server Protocol:

**Developer communities and tools are restructuring to work together and be more effective.**

# Language Server Protocol:

**Desktop editors will keep up better  
with changes in languages.**

# Language Server Protocol:

**Cloud editors are set to significantly improve their functionality.**

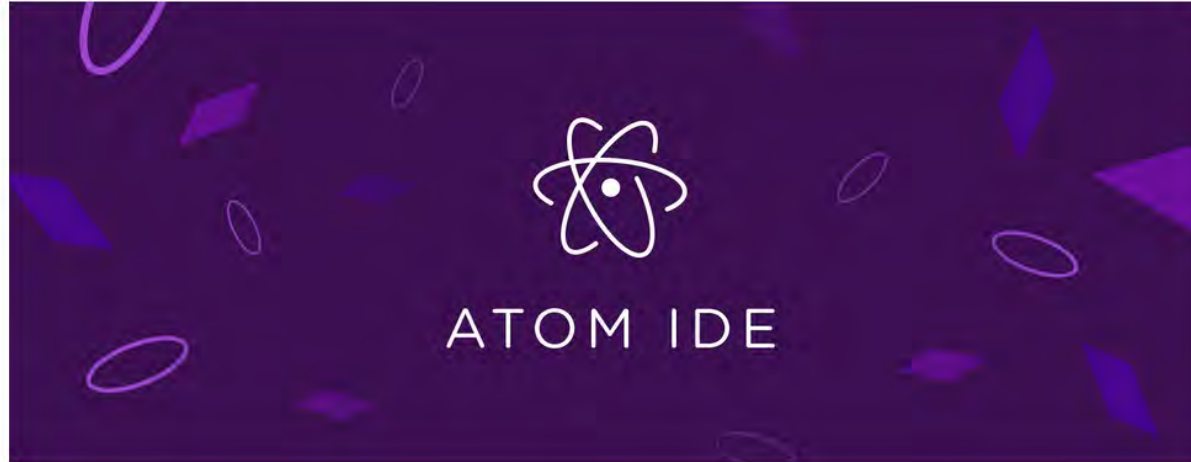
# Github?



## Introducing Atom-IDE

September 12, 2017  damieng

 Tweet



GitHub, [in collaboration with Facebook](#), are pleased to announce the launch of Atom-IDE - a set of optional packages to bring IDE-like functionality to Atom.

The start of this journey includes smarter context-aware auto-completion as well as a host of code navigation features such as an outline view, go to definition, find all references as well as other useful functions such as hover-to-reveal information, errors and warnings (diagnostics) and document formatting.

Our initial release includes packages for TypeScript, Flow, JavaScript, Java, C# and PHP that utilize the power of language servers to provide deep syntactical analysis of your code and projects. The [language server protocol](#) is being adopted by a number of organizations including Microsoft, Eclipse, Sourcegraph, Palantir, Red Hat, Facebook and now GitHub too!

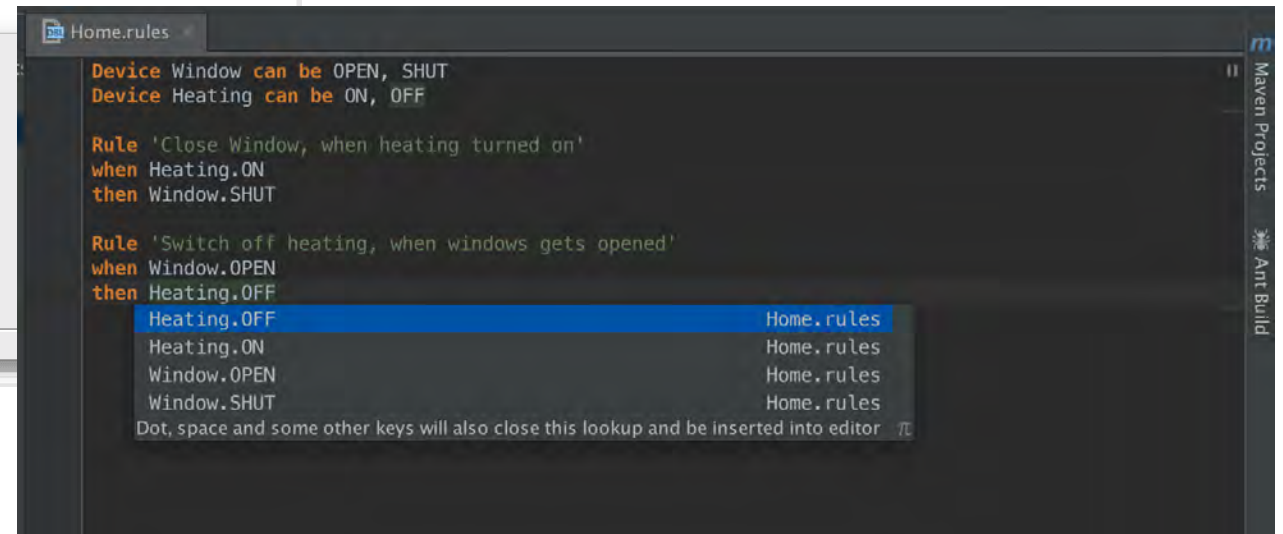
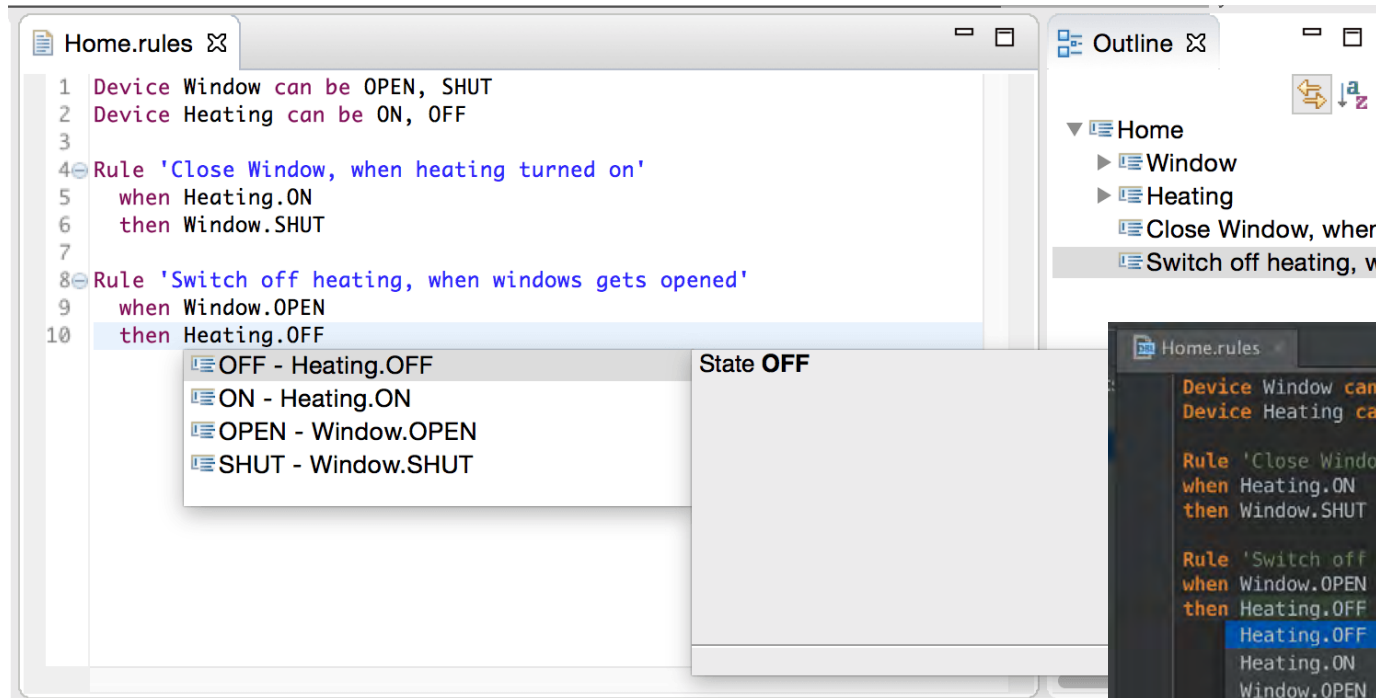
<http://blog.atom.io/2017/09/12/announcing-atom-ide.html>



# Language Server Protocol:

**Let the fragmentation continue...**

# Every developer deserves their very own domain specific language...



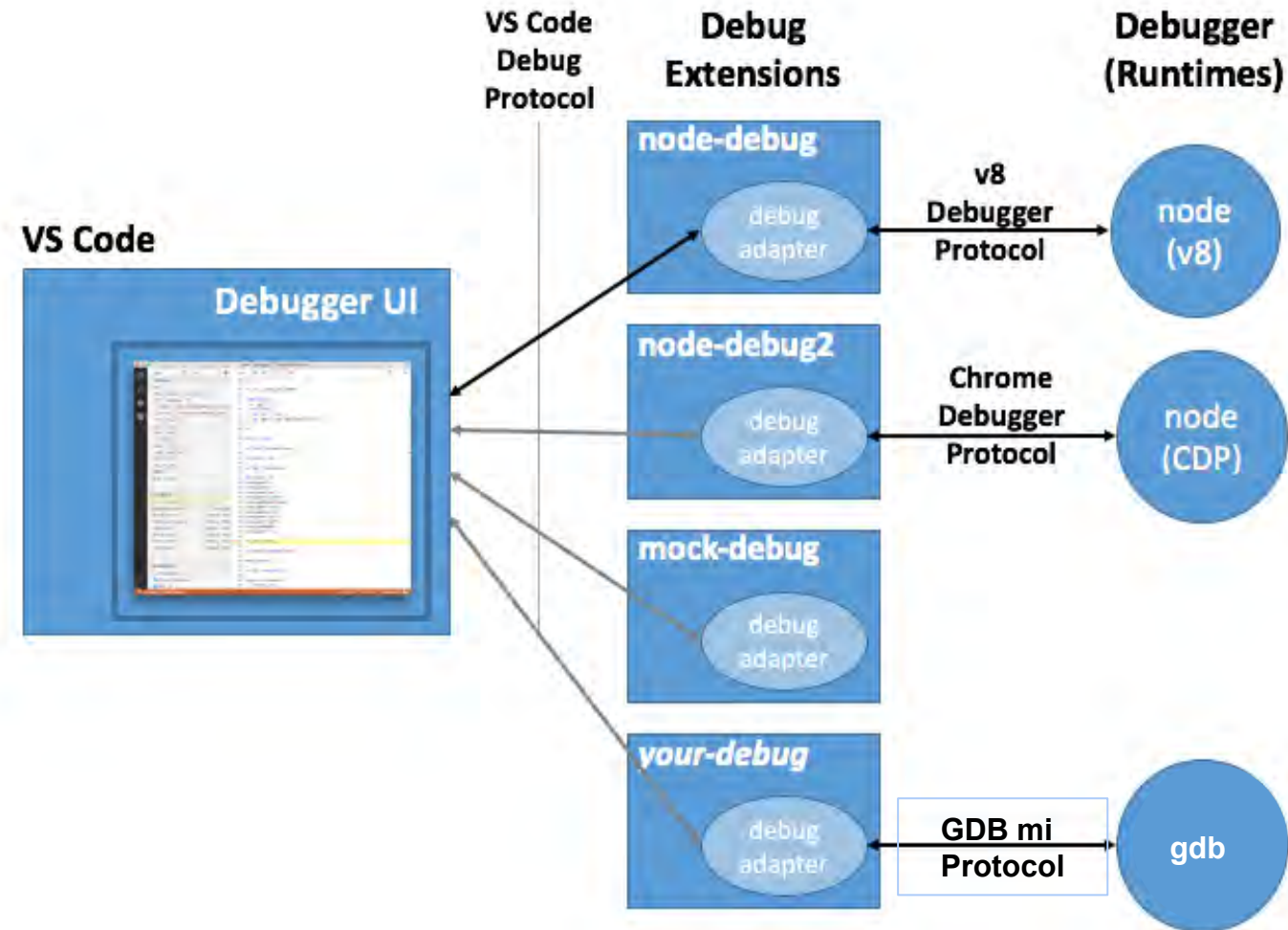
[www.eclipse.org/xtext](http://www.eclipse.org/xtext)



Tracy Miranda, @tracymiranda

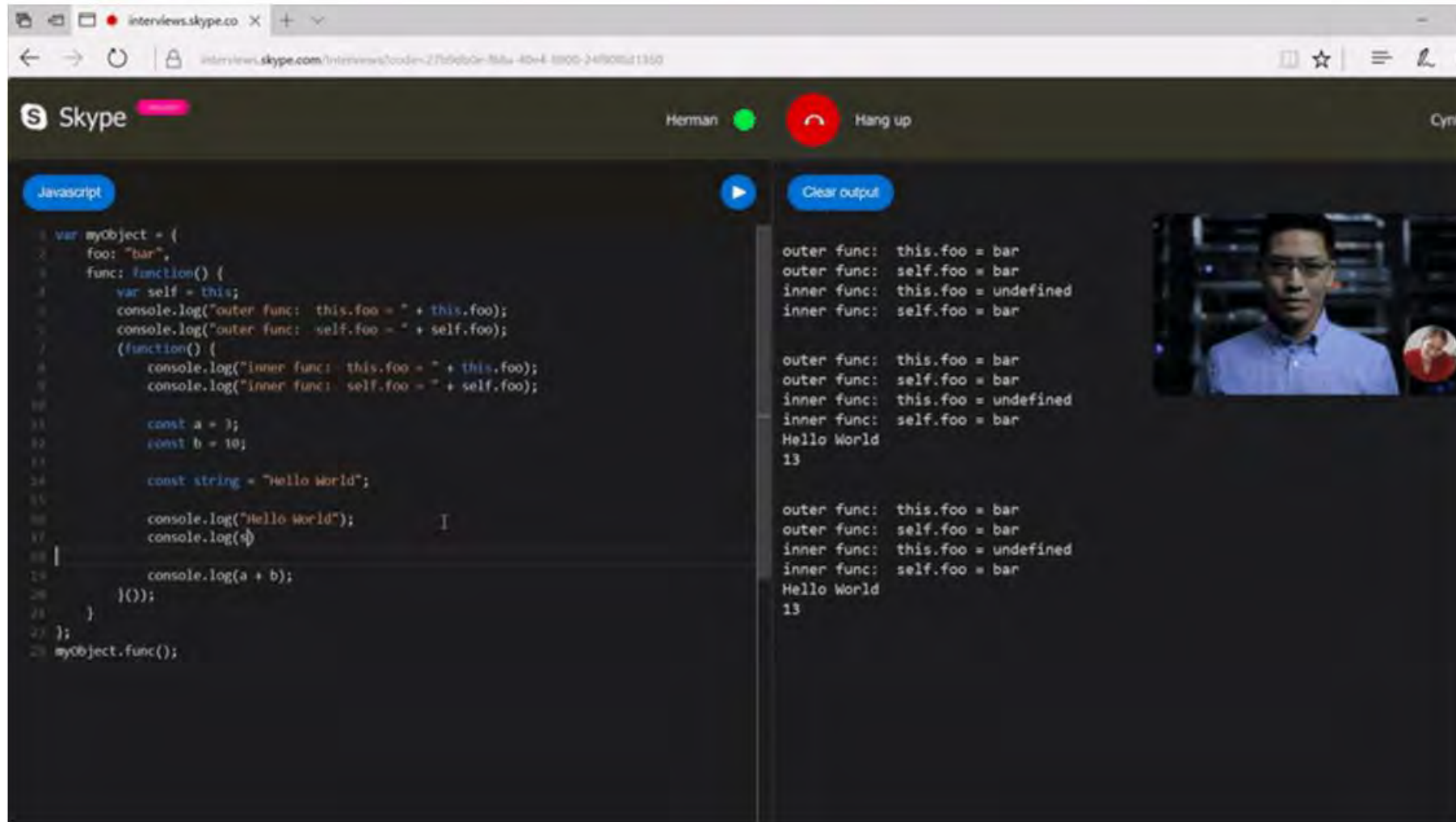


# Debug Protocol



**The future is distributed:  
Monoliths to microservices.  
IDEs to micro-ides**

# Lots of new possibilities...



The screenshot shows a Skype interview window with a code editor on the left and a console on the right. The code editor contains the following JavaScript code:

```
1 var myObject = {  
2   foo: "bar",  
3   func: function() {  
4     var self = this;  
5     console.log("outer func: this.foo = " + this.foo);  
6     console.log("outer func: self.foo = " + self.foo);  
7     (function() {  
8       console.log("inner func: this.foo = " + this.foo);  
9       console.log("inner func: self.foo = " + self.foo);  
10  
11     const a = 3;  
12     const b = 10;  
13  
14     const string = "Hello World";  
15  
16     console.log("Hello World");  
17     console.log(4);  
18  
19     console.log(a + b);  
20   }());  
21 }  
22 };  
23 myObject.func();
```

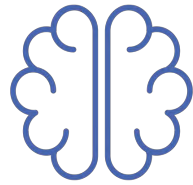
The console output shows the following log messages:

```
outer func: this.foo = bar  
outer func: self.foo = bar  
inner func: this.foo = undefined  
inner func: self.foo = bar  
  
outer func: this.foo = bar  
outer func: self.foo = bar  
inner func: this.foo = undefined  
inner func: self.foo = bar  
Hello World  
13  
  
outer func: this.foo = bar  
outer func: self.foo = bar  
inner func: this.foo = undefined  
inner func: self.foo = bar  
Hello World  
13
```

<https://www.producthunt.com/posts/interviews-on-skype>



**VISUAL (HIGHER LEVEL)**



**ARTIFICIAL INTELLIGENCE**



**CLOUD**

# Gen Z Codes in the Cloud by Default

The screenshot displays the Microsoft MakeCode IDE interface for a micro:bit project. The top navigation bar includes 'micro:bit', 'Projects', 'Share', 'Blocks', and 'JavaScript' tabs, along with a search icon, a settings gear, and the Microsoft logo. On the left, a virtual micro:bit board is shown with pins labeled 0, 1, 2, 3V, and GND. Below the board are icons for running, refreshing, and deleting. A central sidebar contains a search bar and a categorized menu: Basic, Input, Music, Led, Radio (highlighted), More, Loops, Logic, Variables, Math, Advanced, Functions, Arrays, Text, and Game. The main workspace shows a block-based code editor with the following blocks: 'radio send number' (0), 'radio send value' (name, 0), 'radio send string' (''), 'on radio received' (receivedNumber), 'on radio received' (name, value), 'on radio received' (receivedString), and 'radio set group' (1). A tooltip for 'radio set group' explains: 'Sets the group id for radio communications. A micro:bit can only listen to one group ID at any time.' Below the code editor is a 'show leds' block. At the bottom, there is a 'Download' button, a file name 'Untitled', and navigation controls.

# Shape The Future

**Design, design, design**

**Integrate all the things**

**Think extensible**

**Default to open source**