



Incremental Graph Code (IGC)

Bridging Incremental Programming and Complex Software Development in
Modern Programming Environments

Max Boksem, Thomas van Binsbergen



Background

Typical Software Development Environments

Background

Text-based Integrated Development Environment (1)

```
IncrGraph > backend > src > routes > file-explorer.ts > router.delete("/") ca
857 router.delete("/session", async (req: Request, res: Response) => {
862   return res
863     .status(400)
864     .json({ error: "File path and session ID are required" });
865 }
866
867 // Get the session directory
868 const sessionDir = path.join(
869   path.dirname(filePath),
870   ".sessions",
871   path.basename(filePath),
872   sessionId,
873 );
874
875 // Remove the session directory
876 await fs.remove(sessionDir);
877
878 // Check if the primary session is affected
879 const sessionsConfigPath = path.join(
880   path.dirname(filePath),
881   ".sessions",
882   path.basename(filePath),
883   "session.config.json",
884 );
885 const sessionsConfigData = await fs.readJSON(sessionsConfigPath)
886 if (sessionsConfigData.current === sessionId) {
887   // Find the most current session
888   // Go through each session config file and find the one with
889   let mostRecentSession = "";
890   let mostRecentTimestamp = 0;
891   const sessionDirs = await getSubDirectories(
892     path.join(
893       path.dirname(filePath),
894       ".sessions",
895       path.basename(filePath),
896     ),
897   );
898   for (const session of sessionDirs) {
899     const sessionConfigPath = path.join(
```

```
VIM - Vi IMproved
version 9.0.2142
by Bram Moolenaar et al.
Vim is open source and freely distributable

Become a registered Vim user!
type :help register<Enter> for information

type :q<Enter> to exit
type :help<Enter> or <F1> for on-line help
type :help version9<Enter> for version info
```

Background

Text-based

The image shows a development environment with a code editor on the left and a graph visualization tool on the right. The code editor displays C++ code for a document parser, including a `DocPara` class. The graph tool, labeled 'Code Atlas', visualizes the relationships between code elements like `DocNode`, `DocPara`, and `DocParBlock`. A search bar at the bottom of the graph tool shows the results for 'doc node hierarchy'.

```

1103     void accept(DocVisitor* v) { CompAcc
1104     bool hasInOutSpecifier() const { ret
1105     bool hasTypeSpecifier() const { ret
1106
1107     private:
1108     Type m_type;
1109     bool m_hasInOutSpecifier;
1110     bool m_hasTypeSpecifier;
1111 };
1112
1113 /** Node representing a paragraph in the
1114 class DocPara : public CompAccept<DocPar
1115 {
1116 public:
1117 DocPara(DocNode* parent) :
1118     m_isFirst(FALSE), m_isLast(
1119     int parse();
1120     Kind kind() const { return
1121     bool isEmpty() const { return
1122     void accept(DocVisitor* v) { CompAc
1123     void markFirst(bool v=TRUE) { m_isFi
1124     void markLast(bool v=TRUE) { m_isLa
1125     bool isFirst() const { return
  
```

Graph visualization details:

- Nodes: `DocNode`, `DocPara`, `DocParBlock`, `~DocNode`, `parse`, `writeSyno`, `parse`, `handleHtn`, `Tag`, `handleHtn`, `handleCor`, `m_isFirst`, `pop`, `markFirst`, `push`, `append`.
- Relationships: `DocNode` is a base class for `DocPara` and `DocParBlock`. `DocPara` has a `parent` of type `DocNode`. `DocPara` has a `parse` method that calls `accept`, `kind`, and `DocParBlock`. `DocParBlock` has a `parse` method that calls `pop`, `markFirst`, `push`, and `append`.

Search results for 'doc node hierarchy':

Name	Add/Replace Graph
doc node hierarchy	Show Graph
	Delete Graph

Filter: parse code, doc node hierarchy

at (1)

utable
nation
ne help
on info

Background

Incremental Programming Environment (IPE)

```
Python 3.12.5 (main, Aug 6 2024, 19:08:49) [Clang 15.0.0 (clang-1500.0.0)]
Type "help", "copyright", "credits" or "license" for more information.
>>> x = 5
>>> print(x)
5
>>> █
```

The screenshot shows a Jupyter Notebook interface. At the top, it says "jupyter How to use Jupyter Notebook Last Checkpoint: Last Wednesday at 10:23 (unsaved changes)". The interface includes a menu bar with "File", "Edit", "View", "Insert", "Cell", "Kernel", "Widgets", and "Help". Below the menu bar is a toolbar with icons for file operations and execution. The notebook content is as follows:

- THIS IS THE FIRST TEST**
- In [2]: `print('Hello, World!')`
`print('This is just nice!')`
Hello, World!
This is just nice!
- In [3]: `import pandas as pd`
- In []: `pd.read_csv`
- THIS IS A MARKDOWN CELL**
- In [4]: `#this is a test cell`
`first_int = 1`
`first_string = 'hey'`
- In [5]: `print(first_int)`
1

Design

How can we bridge these two environments?

Complexity management tools from typical IDEs

Incremental Programming techniques from IPEs



Design

How can we bridge these two environments?

Integrated Development Environments

Incremental Programming Environments

Design

How can we bridge these two environments?

Integrated Development Environments

- Manage large, multi-file codebases
- Strong project structure (files, packages, classes)
- Readability and code comprehension
- Extensibility
- Integration with version control

Incremental Programming Environments

Design

How can we bridge these two environments?

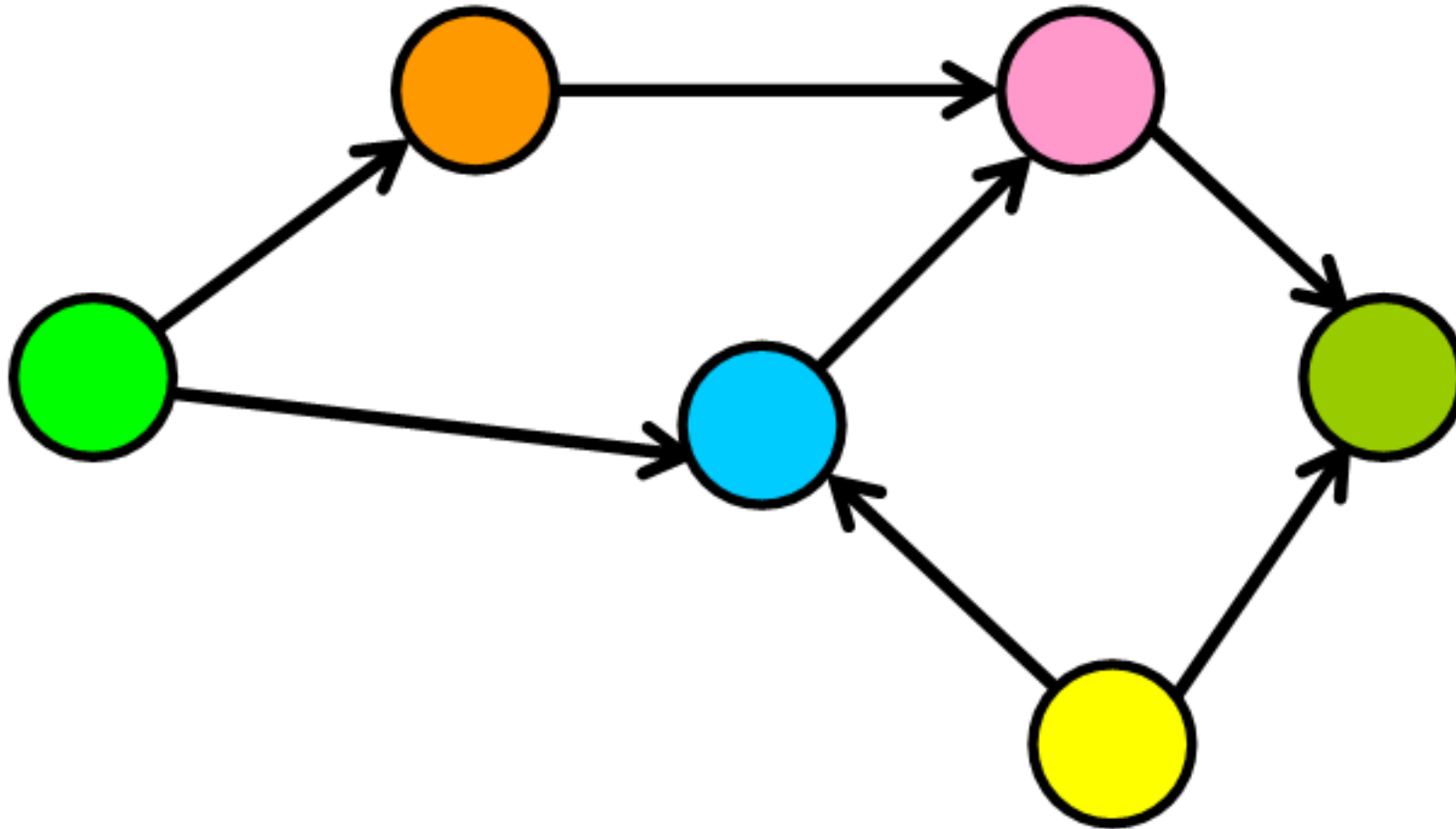
Integrated Development Environments

- Manage large, multi-file codebases
- Strong project structure (files, packages, classes)
- Readability and code comprehension
- Extensibility
- Integration with version control

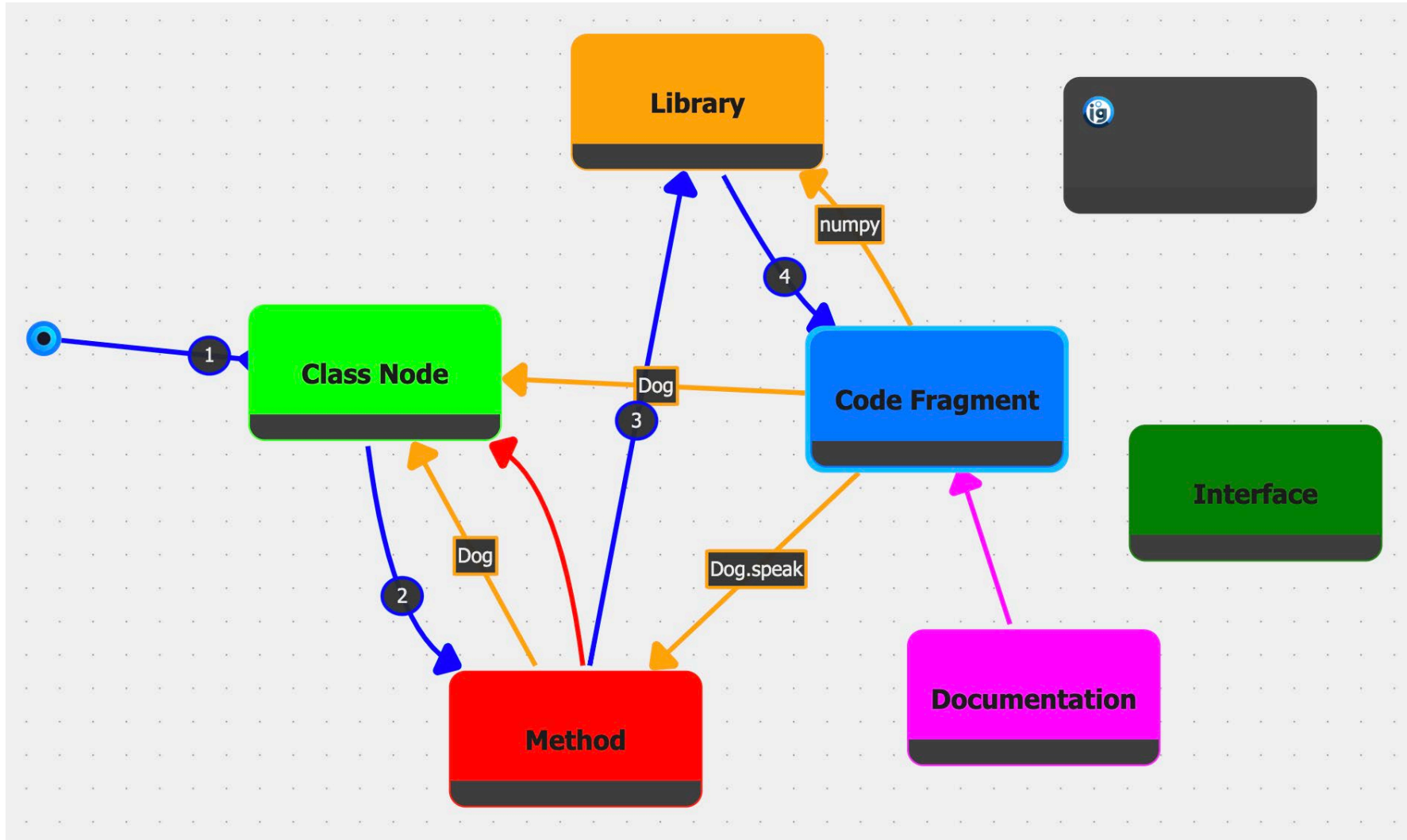
Incremental Programming Environments

- Run and test small code snippets quickly
- Immediate feedback and rapid iteration
- Exploratory coding approach
- Documentation
- Incremental recomputation

Design



Design





Incremental Graph Code (IncrCode)
FILE EDIT PROJECT INFO

content

FirstFile.igc

Session Configuration

START NEW SESSION

Select Session

IGC_09-12-2024_11:29:52

```

{ 3 items
  "Dog" : string "<class>"
  "numpy" : string "<module>"
  "d" : string "<Dog>"
}

```

DELETE SESSION

Graph Editor

Code Editor

CODE NODE VIEW PROJECTION VIEW

Initializer

Initializes the **Class** and **Library** code.

```

1 import numpy
2
3 d = Dog()
4 d.speak()

```

OUTPUT ERRORS CONFIGURATION METRICS

Woof !

Node

Code Fragment Node

Name

Code Fragment

DELETE

Incremental Graph Code React Flow

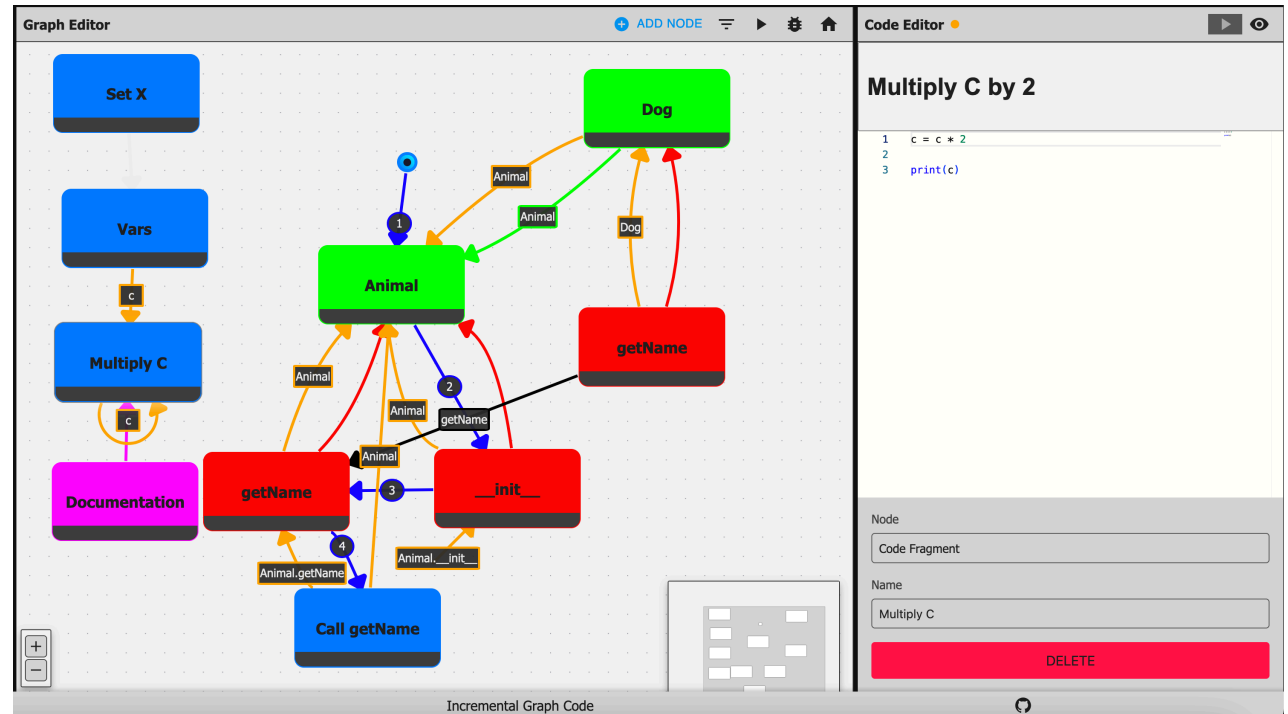
Incremental Graph Code (IGC)

Overview

- Electron Webapp
 - Frontend – React (Typescript) using ReactFlow
 - Responsible for visuals, logic, and overall operation
 - Backend – Node.js (Typescript)
 - Handles file operations and code analysis

Currently Supports Python

Open-Source and Community focused



Case Studies

- Computational Notebook
- Projectional Display (*PescaJ Paint 2023*)
- Exploratory Programming View
- Composition / Architectural View

Final Thoughts

Max Boksem – max.boksem.nl@gmail.com

Thomas van Binsbergen – l.t.vanbinsbergen@uva.nl

Github

