Writing Code the “Right” Way: Empowering Developers to Understand Code

Thomas LaToza

8/10/2022
4
Understanding code
Your AI pair programmer

GitHub Copilot uses the OpenAI Codex to suggest code and entire functions in real-time, right from your editor.

```
#!/usr/bin/env ts-node

import { fetch } from "fetch-h2";

// Determine whether the sentiment of text is positive
// Use a web service
async function isPositive(text: string): Promise<boolean> {
  const response = await fetch('http://text-processing.com/api/sentiment/', {
    method: "POST",
    body: `text=${text}`,
    headers: {
      "Content-Type": "application/x-www-form-urlencoded",
    },
  });
  const json = await response.json();
  return json.label === "pos";
}
```
Rationale (42)
Why was it done this way? (14) [15][7]
Why wasn’t it done this other way? (15)
Was this intentional, accidental, or a hack? (9)[15]
How did this ever work? (4)

Debugging (26)
How did this runtime state occur? (12) [15]
What runtime state changed when this executed? (2)
Where was this variable last changed? (1)
How is this object different from that object? (1)
Why didn’t this happen? (3)
How do I debug this bug in this environment? (3)
In what circumstances does this bug occur? (3) [15]
Which team’s component caused this bug? (1)

Intent and Implementation (32)
What is the intent of this code? (12) [15]
What does this do (6) in this case (10)? (16) [24]
How does it implement this behavior? (4) [24]

Refactoring (25)
Is there functionality or code that could be refactored? (4)
Is the existing design a good design? (2)
Is it possible to refactor this? (9)
How can I refactor this (2) without breaking existing users(7)? (9)
Should I refactor this? (1)
Are the benefits of this refactoring worth the time investment? (3)

History (23)
When, how, by whom, and why was this code changed or inserted? (13)[7]
What else changed when this code was changed or inserted? (2)
How has it changed over time? (4)[7]
Has this code always been this way? (2)
What recent changes have been made? (1)[15][7]
Have changes in another branch been integrated into this branch? (1)

Implications (21)
What are the implications of this change for (5) API clients (5), security (3), concurrency (3), performance (2), platforms (1), tests (1), or obfuscation (1)? (21) [15][24]

Testing (20)
Is this code correct? (6) [15]
How can I test this code or functionality? (9)
Is this tested? (3)
Is the test or code responsible for this test failure? (1)
Is the documentation wrong, or is the code wrong? (1)

Implementing (19)
How do I implement this (8), given this constraint (2)? (10)
Which function or object should I pick? (2)
What’s the best design for implementing this? (7)

Control flow (19)
In what situations or user scenarios is this called? (3) [15][24]
What parameter values does each situation pass to this method? (1)
What parameter values could lead to this case? (1)
What are the possible actual methods called by dynamic dispatch here? (6)
How do calls flow across process boundaries? (1)
How many recursive calls happen during this operation? (1)
Is this method or code path called frequently, or is it dead? (4)
What throws this exception? (1)
What is catching this exception? (1)

Contracts (17)
What assumptions about preconditions does this code make? (5)
What assumptions about post(3)/pre(2)conditions can be made?
What exceptions or errors can this method generate? (2)
What are the constraints on or normal values of this variable? (2)
What is the correct order for calling these methods or initializing these objects? (2)
What is responsible for updating this field? (1)

Performance (16)
What is the performance of this code (5) on a large, real dataset (3)? (8)
Which part of this code takes the most time? (4)
Can this method have high stack consumption from recursion? (1)
How big is this in memory? (2)
How many of these objects get created? (1)

Teammates (16)
Who is the owner or expert for this code? (3)[7]
How do I convince my teammates to do this the “right way”? (12)
Did my teammates do this? (1)

Policies (15)
What is the policy for doing this? (10) [24]
Is this the correct policy for doing this? (2) [15]
How is the allocation lifetime of this object maintained? (3)

Type relationships (15)
What are the composition, ownership, or usage relationships of this type? (5) [24]
What is this type’s type hierarchy? (4) [24]
What implements this interface? (4) [24]
Where is this method overridden? (2)

Data flow (14)
What is the original source of this data? (2) [15]
What code directly or indirectly uses this data? (5)
Where is the data referenced by this variable modified? (2)
Where can this global variable be changed? (1)
Where is this data structure used (1) for this purpose? (1) [24]
What parts of this data structure are modified by this code? (1) [24]
What resources is this code using? (1)

Location (13)
Where is this functionality implemented? (5) [24]
Is this functionality already implemented? (5) [15]
Where is this defined? (3)

Building and branching (11)
Should I branch or code against the main branch? (1)
How can I move this code to this branch? (1)
What do I need to include to build this? (8)
What includes are unnecessary? (2)
How do I build this without doing a full build? (1)
Why did the build break? (2)[50]
Which preprocessor definitions were active when this was built? (1)

Architecture (11)
How does this code interact with libraries? (4)
What is the architecture of the code base? (3)
How is this functionality organized into layers? (1)
Is our API understandable and flexible? (5)

Concurrency (9)
What threads reach this code (4) or data structure (2)? (6)
Is this class or method thread-safe? (2)
What members of this class does this lock protect? (1)

Dependencies (5)
What depends on this code or design decision? (4)[7]
What does this code depend on? (1)

Method properties (2)
How big is this code? (1)
How overloaded are the parameters to this function? (1)
What does this do?
What does these functions do?

What does this do in this case?
What happens if an exception is thrown?
What happens if this operation times out?
What happens if the remote service is slow?

What is the intent of the code?
What is it trying to accomplish?

How does it implement this behavior?
How is this data aggregated and how is it translated from one place to another.
How does this class (or collection of classes) fulfill the functional feature of the application?
What depends on this code or design decision?
What else depends on this code?
Who else uses this code / function. (i.e. If we change this, what will break simply because someone else has found a way to use this and we don't even know they are doing so...)

What are the implications of this change for API clients, security, concurrency, performance, platforms, tests, or obfuscation?
What is the implication of these changes in terms of the backward compatibility?
Across components with a code base the size of complete applications, what are the implications of a functional change in base storage to all accessors in the system (including clients of applications built on top of the place where the change is occurring)

How can I refactor this without breaking existing users?
How can I refactor this piece w/o causing an avalanche of new places to refactor?
Why was it done this way?
Why was this code structured in this way?
Why was this done this way? Is there some reason for this ancient code doing what it does that I'm missing?

Why wasn't it done this other way?
Why didn’t they use this method/object/interface as it appears to have been designed?
Why did the original developer not use library function X? (was there a good reason or just ignorance)

Was this intentional, accidental, or a hack?
Is the lack of parameter validation (most often lack of null checks) intentional or incidental?
Is the lack of "sealed" on the class intentional or incidental? If intentional, why? (assuming no virtual methods are present).

Alaboudi, LaToza. Edit-run behavior in programming and debugging. VL/HCC 2021. (best paper)
Alaboudi, LaToza. Edit-run behavior in programming and debugging. VL/HCC 2021. (best paper)
The edit step lasted for one minute on average and was twice as long in programming as in debugging. The run step lasted for about half a minute for both programming and debugging.

Edit steps occupied the majority of edit-run cycle time in debugging and programming.

During each edit step, developers on average edited only one file.

Alaboudi, LaToza. Edit-run behavior in programming and debugging. VL/HCC 2021. (best paper)
Alaboudi, LaToza. Edit-run behavior in programming and debugging. VL/HCC 2021. (best paper)
**JEditBuffer**

- `fireTransactionComplete`
- `getFoldLevel`

**BufferHandler**

- `transactionComplete`
- `doDelayedUpdate`

getFoldLevel updates fold data structure

getFoldLevel has effects

getFoldLevel fires events

Folds updated by buffer changes are updated on screen

Buffer mutating operations result in doDelayedUpdate call

Folds are initialized at startup

isFoldStart calls getFoldLevel at startup

getFoldLevel determines if folds must be set

**CRITIQUED**

doDelayedUpdate triggers update

doDelayedUpdate does changes that happen later

Many methods call getFoldLevel

getFoldLevel is mutually recursive with FoldHandler.getFoldLevel

When does this happen? What does this do?

Reacher

How do I use this function? Why is it this way? Why can't it be this other way?
programming work
public void beforeAllMethods(ICompilationUnit compUnit, CompilationUnit rootNode) {
    if (project == null || !project.equals(compUnit.getJavaProject())) {
        // we have a new project: reset the type hierarchy
        project = compUnit.getJavaProject();
        try {
            IProgressMonitor monitor = getInput().getProgressMonitor().isNone() ? null : getInput();
            types = new CachedTypeHierarchy(project, monitor);
            FreeVars.setHierarchy(types);
            retriever.retrieveRelationships(ResourcesPlugin.getWorkspace().getRoot(), types);

            FusionAnalysis
                + beforeAllMethods(.....)
                + runAnalysis(.....)

            XMLRetriever
                + retrieveRelationships(.....)
                + getStartContext(.....)

            AbstractCrystalMethodAnalysis
        }
    }
}
“So basically, what I’m trying to do is, we create some tables and we don’t want to create them by default now. So I have to figure out where the code is assuming that the tables are already there. So it’s not very clear where that happens.”
<table>
<thead>
<tr>
<th>Longest investigation activities</th>
<th>Time (mins)</th>
</tr>
</thead>
<tbody>
<tr>
<td>How is this data structure being mutated in this code?</td>
<td>83</td>
</tr>
<tr>
<td>“Where [is] the code assuming that the tables are already there?”</td>
<td>53</td>
</tr>
<tr>
<td>How [does] application state change when $m$ is called denoting startup completion?</td>
<td>50</td>
</tr>
<tr>
<td>What decisions might be incompatible with reuse in new context?</td>
<td>24</td>
</tr>
<tr>
<td>“Is [there] another reason why $status$ could be non-zero?”</td>
<td>11</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Longest debugging activities</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Where is method $m$ generating an error?</td>
<td>66</td>
</tr>
<tr>
<td>What resources are being acquired to cause this deadlock?</td>
<td>51</td>
</tr>
<tr>
<td>“When they have this attribute, they must use it somewhere to generate the content, so where is it?”</td>
<td>35</td>
</tr>
<tr>
<td>“What [is] the test doing which is different from what my app is doing?”</td>
<td>30</td>
</tr>
<tr>
<td>How are these thread pools interacting?</td>
<td>19</td>
</tr>
</tbody>
</table>
A search along **feasible paths** downstream or upstream from a statement \((m)\) for **target statements** matching search criteria (calls to method \(e\))

LaToza, Myers. Developers ask reachability questions. ICSE 2010.
Reachability questions
asked > 9x a day
82% agree can be hard to answer
Not easier with increasing codebase knowledge or experience

1. What are the implications of this change? (e.g., what might break)
2. How does application behavior vary in these different situations that might occur?
3. Could this method call potentially be slow in some situation I need to consider?
4. To move this functionality (e.g., lines of code, methods, files) to here, what else needs to be moved?
5. Is this method call now redundant or unnecessary in this situation?
6. Across this path of calls or set of classes, where should functionality for this case be inserted?
7. When investigating some application feature or functionality, how is it implemented?
8. In what situations is this method called?
9. What is the correct way to use or access this data structure?
10. How is control getting (from that method) to this method?
11. What parts of this data structure are accessed in this code?
12. How are instances of these classes or data structures created and assembled?

LaToza, Myers. Developers ask reachability questions. ICSE 2010.
LaToza, Myers. Developers ask reachability questions. ICSE 2010.
A search along **feasible paths** downstream or upstream from a statement \((m)\) for **target statements** matching search criteria (calls to method \(e\))
Developers with REACHER were 5.6 times more successful than those working with Eclipse only.

Find: `Usages of initialCapacity(int) in All Places`

### Method
- `initialCapacity(int)`

### Found usages 31 usages

### Unclassified usage 31 usages

- **guava 1 usage**
  - `com.google.common.cache 1 usage`
    - `CacheBuilderSpec 1 usage`
      - `toCacheBuilder() 1 usage`
      - 170 `builder.initialCapacity(initialCapacity);`

- **guava-tests 30 usages**
  - `com.google.common.cache 30 usages`
    - `CacheBuilderFactory 1 usage`
      - `createCacheBuilder(Integer, Integer, Integer, DurationSpec, DurationSpec, DurationSpec, Strength, ...) 1 usage`
      - 175 `builder.initialCapacity(initialCapacity);`
    - `CacheBuilderSpecTest 4 usages`
      - `testParse_initialCapacity() 1 usage`
      - 59 `CacheBuilder.newBuilder().initialCapacity(10), CacheBuilder.from(spec));`
      - `testParse_multipleKeys() 1 usage`
      - 387 `.initialCapacity(10)`
      - `testParse_whitespaceNotAllowed() 1 usage`
      - 413 `.initialCapacity(10)`
      - `testCacheBuilderFrom_string() 1 usage`
      - 516 `.initialCapacity(10)`
    - `CacheBuilderTest 7 usages`
      - `testInitialCapacity_negative(1 1 usage`
      - 66 `builder.initialCapacity(-1));`
      - `testInitialCapacity_setTwice() 2 usages`
      - 73 `CacheBuilder<Object, Object> builder = CacheBuilder.newBuilder().initialCapacity(16);`
      - 76 `builder.initialCapacity(16);`
      - `testInitialCapacity_small() 1 usage`
      - 84 `LoadingCache<?, ?> cache = CacheBuilder.newBuilder().initialCapacity(5).build(IdentityLoader());`
      - `testInitialCapacity_smallest() 1 usage`
      - 96 `LoadingCache<?, ?> cache = CacheBuilder.newBuilder().initialCapacity(0).build(IdentityLoader());`
      - `testInitialCapacity_large() 1 usage`
      - 108 `CacheBuilder.newBuilder().initialCapacity(Integer.MAX_VALUE);`
      - `testMaximumSize_largerThanInt() 1 usage`
      - 177 `CacheBuilder.newBuilder().initialCapacity(512).maximumSize(Long.MAX_VALUE);`
    - `LocalCacheTest 17 usages`
Find Unique Usages

<table>
<thead>
<tr>
<th>Time (mins)</th>
<th>Baseline</th>
<th>Find Unique Usages</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>33</td>
<td>21</td>
</tr>
<tr>
<td>8.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>34</td>
<td></td>
<td></td>
</tr>
<tr>
<td>This is a serious problem for me</td>
<td>% agree</td>
<td></td>
</tr>
<tr>
<td>----------------------------------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td><strong>Code Understanding</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Understanding the rationale behind a piece of code</strong></td>
<td>66%</td>
<td></td>
</tr>
<tr>
<td>Understanding code that someone else wrote</td>
<td>56%</td>
<td></td>
</tr>
<tr>
<td>Understanding the history of a piece of code</td>
<td>51%</td>
<td></td>
</tr>
<tr>
<td>Understanding code that I wrote a while ago</td>
<td>17%</td>
<td></td>
</tr>
<tr>
<td><strong>Task Switching</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Having to switch tasks often because of requests from my teammates or manager</td>
<td>62%</td>
<td></td>
</tr>
<tr>
<td>Having to switch tasks because my current task gets blocked</td>
<td>50%</td>
<td></td>
</tr>
<tr>
<td><strong>Modularity</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Being aware of changes to code elsewhere that impact my code</td>
<td>61%</td>
<td></td>
</tr>
<tr>
<td>Understanding the impact of changes I make on code elsewhere</td>
<td>55%</td>
<td></td>
</tr>
<tr>
<td><strong>Links between Artifacts</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Finding all the places code has been duplicated</td>
<td>59%</td>
<td></td>
</tr>
<tr>
<td>Understanding who “owns” a piece of code</td>
<td>50%</td>
<td></td>
</tr>
</tbody>
</table>

**Rationale (42): Why wasn’t it done this other way? (15)**

Intent and implementation (32): What does this do (6) in this case (10)? (16)

Debugging (26): How did this runtime state occur? (12)

Refactoring (25): How can I refactor this (2) without breaking existing users(7)? (9)

History (23): Who, when, how, and why was this code changed or inserted? (13)
Because of sharding, interactions between Artifact should be indirected through Command classes.

Why??
Init should be called on FooBar!

Here's why:

Documentation

No sharding is used in this application.

Documentation

Because of sharding, interactions between Artifact should be indirected through Command classes.
Because of sharding, interactions between Artifact should be indireced through Command classes.
Because of sharding, interactions between Artifact should be indirected through Command classes.
Because of sharding, interactions between Artifact should be indireced through Command classes.
Include PRs with Change Requests
31685 PRs
Remove potentially useless PRs
795 PRs
Collect review comments
493 PRs
Remove useless RCs
2343 RCs
Remove useless RCs
1323 RCs
Open Coding

31685 PRs
795 PRs
493 PRs
2343 RCs
1323 RCs

PAT Classification
Defect Classification

“please add a target="_blank" ... to all the external links”

IF there's an external link, THEN must have target="_blank"

<table>
<thead>
<tr>
<th>Category</th>
<th>Defects Found</th>
</tr>
</thead>
<tbody>
<tr>
<td>AST Pattern Checkers</td>
<td>22.14%</td>
</tr>
<tr>
<td>Style checkers</td>
<td>18.7%</td>
</tr>
<tr>
<td>Continuous Integration Tools</td>
<td>4.66%</td>
</tr>
<tr>
<td>Architectural Style Checkers</td>
<td>3.97%</td>
</tr>
<tr>
<td>Data Flow Analyzers</td>
<td>4.27%</td>
</tr>
<tr>
<td>Dead Code Detectors</td>
<td>2.06%</td>
</tr>
<tr>
<td>Code Clone Detectors</td>
<td>1.83%</td>
</tr>
<tr>
<td>Compilers</td>
<td>1.53%</td>
</tr>
<tr>
<td>Test Suite Checkers</td>
<td>2.21%</td>
</tr>
<tr>
<td>String Compilers</td>
<td>1.15%</td>
</tr>
<tr>
<td>Non-detectable</td>
<td>37.11%</td>
</tr>
</tbody>
</table>

Mehrpour, LaToza, Kindi. Active documentation: Helping developers follow design decisions. VL/HCC 2019.
package com.crowdcoding.commands;
import com.crowdcoding.entities.artifacts.DesignDoc;
import com.crowdcoding.servlets.ThreadContext;

public abstract class DesignDocCommand extends Command {

protected long DesignDocId;

// This function is called when a new DesignDoc must be created.
public DesignDocCommand create(String title, String description, boolean isApiArtifact) {
    return null;
}

private DesignDocCommand(long DesignDocId) {
    this.DesignDocId = DesignDocId;
    queueCommand(this);
}

// All constructors for DesignDocCommand MUST call queueCommand and the end of
// the constructor to add the
// command to the queue.
private static void queueCommand(Command command) {
    ThreadContext threadContext = ThreadContext.get();
    threadContext.addCommand(command);
}

public void execute(final String projectId) {
    if (DesignDocId != 0) {
        DesignDoc designDoc = DesignDoc.find(DesignDocId);
        if (designDoc == null) {
            System.out.println("error Cannot execute DesignDocCommand. Could not find DesignDocCommand with ID "+ DesignDocId);
        } else {
            execute(designDoc, projectId);
        }
    } else {
        execute(null, projectId);
    }
}

public abstract void execute(DesignDoc designDoc, String projectId);
}
// This function is called when a new DesignDoc must be created.

public static DesignDocCommand create(String title, String description, boolean isApiArtifact) {
    return null;
}

private DesignDocCommand(Long DesignDocId) {
    this.DesignDocId = DesignDocId;
    queueCommand(this);
}

// All constructors for DesignDocCommand MUST call queueCommand and the end of
// the constructor to add the
// command to the queue.

private void queueCommand(Command command) {
    ThreadContext threadContext = ThreadContext.get();
    threadContext.addCommand(command);
}

public void execute(final String projectId) {
    if (DesignDocId != 0) {
        DesignDoc designDoc = DesignDoc.find(DesignDocId);
        if (designDoc == null)
            System.out.println("Error Cannot execute DesignDocCommand. Could not find DesignDoc with id");
        else {
            execute(designDoc, projectId);
        }
    } else {
        execute(null, projectId);
    }
}

public abstract void execute(DesignDoc designDoc, String projectId);

Rules applicable for File:

CrowdCode-master/CrowdCoding/src/com/crowdcoding/commands/DesignDocCommand.java

All Microtask commands must be handled by Command subclasses

If a method is a static method on Command THEN it should implement its behavior by constructing a new Command subclass instance. The Command class contains a number of static methods. Each method creates a specific type of Command by invoking the constructor of the corresponding subclass.

Microtask | Command | Sharding
---|---|---
Examples: 0 out of 54 | Violated: 1 out of 1

Example Snippet for this file

No snippet

Example Snippet for other files

```java
public static ADTCommand create(String description, String name, HashMap<String, String> examples, boolean isApiArtifact) {
    return new Create(description, name, structure, examples, isApiArtifact);
}

public static ADTCommand update(long ADTId, String description, String name, HashMap<String, String> examples) {
    return new Update(ADTId, description, name, structure);
}

public static ADTCommand delete(long ADTId) {
    return new Delete(ADTId);
}

public static TestCommand create(TestDTO test, long functionId, boolean isApiArtifact, boolean isReadOnly) {
    return new Create(test, functionId, isApiArtifact, isReadOnly);
}

public static TestCommand update(TestDTO test) {
```
package com.crowdcoding.commands;

import com.crowdcoding.entities.artifacts.DesignDoc;
import com.crowdcoding.servlets.ThreadContext;

public abstract class DesignDocCommand extends Command {

    protected long DesignDocID;

    // This function is called when a new DesignDoc must be created.
    public static DesignDocCommand create(String title, String description, boolean isApiArtifact)
    return null;

    private DesignDocCommand(long DesignDocId) {
        this.DesignDocId = DesignDocId;
        queueCommand(this);
    }

    // All constructors for DesignDocCommand MUST call queueCommand and the end of
    // the constructor to add the
    // command to the queue.
    private static void queueCommand(Command command) {
        ThreadCommand threadCommand = ThreadContext.getCommand();
        threadContext.addCommand(command);
    }

    public void execute(final String projectId) {
        if (DesignDocId != 0) {
            DesignDoc designDoc = DesignDoc.find(DesignDocId);
            if (designDoc == null)
            else {
                execute(designDoc, projectId);
            }
        } else {
            execute(null, projectId);
}
}

violated rules

All Microtask commands must be handled by Command subclasses
IF a method is a static method on Command THEN it should implement its behavior by
constructing a new Command subclass instance. The Command class contains a number of
static methods. Each method creates a specific type of Command by invoking the constructor
of the corresponding subclass.

DTOs must have a zero-argument constructor
IF a class is a DTO object THEN it needs to have a zero-argument constructor with an empty
body. The Jackson JSON library automatically constructs an instance of DTO objects, using
reflection. To do so, it requires there to be a no-arg constructor that does nothing.

Save() calls should always be committed immediately
IF the save method is called THEN the now() method must be followed immediately. Invoking
the save() method marks an entity to storage to the datastore by Objectify. However, this data
is not yet committed and any subsequent calls to load() will still receive a stale pre-save
version of the data. In order to ensure that all loads() receive the newest version of the data,
every call to save() should be immediately followed by a call to now(), which commits the new
version to the datastore immediately.
Does it reduce time and/or increase chance of success?

How does it change how developers work?
Task

Add a small feature to an existing codebase

- Existing codebase: 9,000 lines of code, 107 Java classes

- Requested code: implement a new class, adding 20 lines of code, editing 2 lines of code

Baseline: same documentation, but as traditional static document
Experimental participants were 3 times faster in starting editing the code and 28% faster in finishing the task.

Participants did not significantly differ in the number lines of code they added or removed.

Experimental participants submitted 98% fewer incorrect LOC.
Control Group

✗ Challenges in finding relevant design decisions within the design documentation

✗ Challenges in connecting code with design decisions

✗ Challenges in finding relevant pieces of code, scattered in different classes

Experimental Group

✓ Used *Violated Rules* page to find relevant design decisions.

✓ Used the violated snippets to identify relevant places to make changes.

✓ Used example snippets listed to compare examples of the rule and the faulty lines of code.

✓ Used real-time feedback to detect errors and violations early, immediately after changing the code without running the application.
Because of sharding, interactions between Artifact should be indirected through Command classes.
If a method is in a public class, then it must be void or have a name which begins with get, search, or find.

```
src:unit//src:block/src:function[ancestor-or-self::src:class[src:specifier[text()="public"]]]
src:unit//src:block/src:function[ancestor-or-self::src:class[src:specifier[text()="public"]]
and (src:type[src:name[text()="void"]] or src:name[starts-with(text(),"get") or starts-with(text(),"search") or starts-with(text(),"find")])]]
```
If a method is in a public class, then it must be void or have a name which begins with get, search, or find.

```java
package src.com.bankapplication.controller

public class

    void get... || search... || find
```
If a method is in a public class, then it must be void or have a name which begins with get, search, or find.

```java
package src.com.bankapplication.controller

public class

void search
```
Properties of public classes in Controller package

Non-void methods are getters, search functions, login, or functions for withdraw or deposit

Controller Assign Tags

Rule must be applied on Specific Files/Folders Add files/folders

cr.com/bankapplication/controller

Step 1: Write the code you want to match in code using the Graphical Editor.

Step 2: Specify what must be true by switching the conditions to 'constraints' by clicking on checkboxes. Constraint elements are highlighted in the Graphical Editor.

Step 3: [Optional] Edit the rule text by adding parentheses and changing 'and' to 'or'.

@annotation
visibility static class className implements Interface extends Superclass

Specify declaration statement

Specify constructor
If a method is in a public class, then it must be void or have a name which begins with get, search, or find.
Specify abstract function

declaration statement of

class
constructor
function

declaration statement

declaration statement is a statement in which a variable is declared by specifying its data type and name. The variable can also be initialized.

@Id private static int ID = 0; //

Submit  Cancel  Clear Form

Properties of public classes in Controller package

Non-void methods are getters, search functions, login, or functions for withdraw or deposit
public static class ClassName implements Interface extends Superclass {

Specify declaration statement

Specify constructor

You can use some regex-like notations for matching values:

- `word` matches value 'word'.
- `pre....` matches values that start with the prefix 'pre' and end with any sequence of characters.
- `....post` matches values that start with any sequence of characters and end with the postfix 'post'.
- `....mid....` matches values that contain the sub-string 'mid'.

To express negation use `!` as follows:

- `!word` matches any value except 'word'.
- `!pre....` matches any value that does NOT start with the prefix 'pre'.
- `!....post` matches any value that does NOT end with the postfix 'post'.
- `!....mid....` matches any value that does NOT contain the sub-string 'mid'.

Values can be mixed with `&` (logical and) and `|` (logical or). For example, `!*pre....mid....` matches values that either do not start with 'pre' OR contain 'mid'.

Design Rule.
\[ DR ::= c \text{ must have } cExp | m \text{ must have } mExp |
    \begin{align*}
    &am \text{ must have } amExp | co \text{ must have } coExp | \\
    &d \text{ must have } dExp | p \text{ must have } pExp
    \end{align*}
\]
\[ c ::= \text{class [with } cExp \text{] [of } c \text{]} \]
\[ m ::= \text{function [with } mExp \text{] [of } c \text{]} \]
\[ am ::= \text{abstract function [with } amExp \text{] [of } c \text{]} \]
\[ co ::= \text{constructor [with } coExp \text{] [of } c \text{]} \]
\[ d ::= \text{declaration statement [with } dExp \text{] [of } (c \mid m \mid co) \text{]} \]
\[ p ::= \text{parameter [with } pExp \text{]} \]
\[ t ::= type [pattern | expr] \]
\[ e ::= \text{extension of } (pattern \mid \text{superclass}) \]
\[ im ::= \text{implementation of } (pattern \mid \text{interface}) \]
\[ ex ::= \text{expression statement [expr] [of } (m \mid co) \text{]} \]
\[ i ::= \text{initial value [expr] [of } d \text{]} \]
\[ r ::= \text{return value [expr]} \]
\[ a ::= \text{annotation [expr]} \]
\[ n ::= \text{name [pattern]} \]
\[ s ::= \text{specifier [pattern]} \]
\[ v ::= \text{visibility [pattern]} \]
\[ cExp ::= (cExp) | cExp \text{ op } cExp | cExp | a | s | v | n | e | im | m | am | co | d | c | r \]
\[ mExp ::= (mExp) | mExp \text{ op } mExp | mExp | a | s | v | t | n | p | r | d | ex \]
\[ amExp ::= (amExp) | amExp \text{ op } amExp | amExp | a | s | v | t | n | p \]
\[ coExp ::= (coExp) | coExp \text{ op } coExp | coExp | a | s | v | p | r | d | ex \]
\[ dExp ::= (dExp) | dExp \text{ op } dExp | dExp | a | s | v | t | n | i \]
\[ pExp ::= (pExp) | pExp \text{ op } pExp | pExp | t | n \]
\[ op ::= \text{and | or} \]
\[ pattern ::= "\{ part ( \&\& | || ) \} part" \]
\[ part ::= [!]\ldots \text{ character}\{\text{character}\} \ldots \]
\[ expr ::= "(character \mid symbol) \{\text{character} \mid symbol\}" \]
baseline (PMD) 4%

our technique 52%
correct query elements
Try it out!

https://devx.cs.gmu.edu/tools/rulepad

Tool Demo
RulePad is implemented as a plugin for IntelliJ IDE. The following demo is a partial implementation of RulePad. Some of RulePad features require IDE APIs, and thus is not available in the online Demo. You can checkout the complete behavior of RulePad here.

```java
public class Test {
    private int counter = 0;

    Test(int j) {
        counter = j;
    }

    public int getCounter() {
        return counter;
    }
}
```
IF a THEN b

a b

f g

f h

a b

k b
IF

THEN

Editable AI

Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead

Cynthia Rudin

Black box machine learning models are currently being used for high-stakes decision making throughout society, causing problems in healthcare, criminal justice and other domains. Some people hope that creating methods for explaining these black box models will alleviate some of the problems, but trying to explain black box models, rather than creating models that are interpretable in the first place, is likely to perpetuate bad practice and can potentially cause great harm to society. The way forward is to design models that are inherently interpretable. This Perspective clarifies the chasm between explaining black boxes and using inherently interpretable models, outlines several key reasons why explainable black boxes should be avoided in high-stakes decisions, identifies challenges to interpretable machine learning, and provides several example applications where interpretable models could potentially replace black box models in criminal justice, healthcare and computer vision.

There has been an increasing trend in healthcare and criminal justice to leverage machine learning (ML) for high-stakes prediction applications that deeply impact human lives. Many of the ML models are black boxes that do not explain their predictions in a way that humans can understand. The lack of transparency and accountability of predictive models can have (and already have) severe consequences; there have been cases of people incorrectly denied parole, poor bail decisions leading to the release of dangerous criminals, ML-based pollution models stating that highly polluted air was safe to breathe and generally poor use of limited valuable resources in criminal justice, medicine, energy reliability, finance and in other domains. Rather than trying to create models that are inherently interpretable, there is a spectrum between fully transparent models (where we understand how all the variables are jointly related to each other) and models that are lightly constrained in model form (such as models that are forced to increase as one of the variables increases, or models that, all else being equal, prefer variables that domain experts have identified as important; see ref. 1). A preliminary version of this manuscript appeared at a workshop, entitled ‘Please stop explaining black box machine learning models for high stakes decisions’.

Key issues with explainable ML

A black box model could be either (1) a function that is too complicated for any human to comprehend or (2) a function that is more complicated than the data it is fitted to. Not all black box models are uninterpretable. A powerful black box model can be designed that is self-consistent; there is a spectrum between fully transparent models (where we understand how all the variables are jointly related to each other) and models that are lightly constrained in model form (such as models that are forced to increase as one of the variables increases, or models that, all else being equal, prefer variables that domain experts have identified as important; see ref. 1).
Interactive Relationships Interface

Current Code Pattern

<table>
<thead>
<tr>
<th>Conditions</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

All Code Patterns

Prioritized:

Standard:

<table>
<thead>
<tr>
<th>#</th>
<th>Conditions</th>
<th>Value</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Parent: &lt;head&gt; AND Element: &lt;meta name</td>
<td>google:category</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Parent: &lt;head&gt; AND Element: &lt;meta name</td>
<td>google:type</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Parent: &lt;head&gt; AND Element: &lt;meta name</td>
<td>google:description</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Parent: &lt;head&gt; AND Element: &lt;meta name</td>
<td>google:title</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Parent: &lt;head&gt; AND Element: &lt;meta name</td>
<td>keywords</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Parent: &lt;head&gt; AND Element: &lt;meta name</td>
<td>theme-color</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Parent: &lt;head&gt; AND Element: &lt;meta content</td>
<td>shopping</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Parent: &lt;head&gt; AND Element: &lt;meta content</td>
<td>website</td>
<td></td>
</tr>
</tbody>
</table>
<div class="theme_alternate">
  <header>
    <h1>StellarGFX</h1>
    <figure>
      <canvas id="draw-header-logo"></canvas>
      <figcaption><h4>Quality GFX Design!</h4></figcaption>
    </figure>
  </header>
</div>

<section class="content" id="sidebar">
  <div class="theme_alternate">
    <h2>Testimonial</h2>
    <figure>
      <blockquote onmouseover="bolden();">
        I couldn't have been more impressed with the images!
      </blockquote>
      <p>Jacob Cecils, Entrepreneur (ai4youth.org)</p>
    </figure>
  </div>
</section>
For a pattern to be valid, at least one condition AND the prediction value must be provided. Fewer conditions will make the pattern general, while more conditions will make it narrowly applicable.

<table>
<thead>
<tr>
<th>Parent Tag</th>
<th>Parent Attribute-Value</th>
<th>Tag</th>
<th>Attribute</th>
<th>➔ Value</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>section</td>
<td>class=headers</td>
<td>img</td>
<td>width</td>
<td>250px</td>
<td></td>
</tr>
</tbody>
</table>
### All Code Patterns

#### Prioritized:

<table>
<thead>
<tr>
<th>#</th>
<th>Conditions</th>
<th>Attribute</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Element: <code>&lt;span</code></td>
<td>style</td>
<td></td>
</tr>
</tbody>
</table>

#### Standard:

<table>
<thead>
<tr>
<th>#</th>
<th>Conditions</th>
<th>Attribute</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Parent: <code>&lt;body</code></td>
<td>class</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Parent: <code>&lt;tag class = &quot;header&quot;&gt; AND Element: </code>&lt;span`</td>
<td>align</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Parent: <code>&lt;tag class = &quot;header&quot;&gt; AND Element: </code>&lt;span`</td>
<td>id</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Element: <code>&lt;meta</code></td>
<td>name</td>
<td></td>
</tr>
</tbody>
</table>

#### Blacklisted:

```html
<div class = "header">
  <span class="content">
    Content
  </span>
</div>
```

```html
<nav>Homepage | More</nav>
```

```html
<span align="left">by John Doe</span>
```
Baseline

Editable AI

### Time (mins)

<table>
<thead>
<tr>
<th>Time (mins)</th>
<th>Baseline</th>
<th>Editable AI</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>57.1</td>
<td>44.1</td>
</tr>
<tr>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td></td>
<td></td>
</tr>
<tr>
<td>45</td>
<td></td>
<td></td>
</tr>
<tr>
<td>60</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### # of Correct Edits

<table>
<thead>
<tr>
<th># of Correct Edits</th>
<th>Baseline</th>
<th>Editable AI</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>11.4</td>
<td>17</td>
</tr>
<tr>
<td>4.25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

p = .03 *

p < .01
“The autocomplete got even more accurate as I voted on the patterns”
Conclusions
programming work
• Empower developers to understand code and make decisions by finding common, hard questions and building tool support around answering questions

• Working with control and data flow is pervasive in understanding code, but still not well supported by today's tools

• Support reuse tasks by helping developers go back and forth between code examples and generalizations of examples, to build trust and explain where recommendations come from

• Encoding developer intent in understandable and editable rules may enable more trust in black-box code generation