Análise de variantes com CodeQL

ISEL – Instituto Superior de Engenharia de Lisboa
Rua Conselheiro Emídio Navarro, 1 | 1959-007 Lisboa

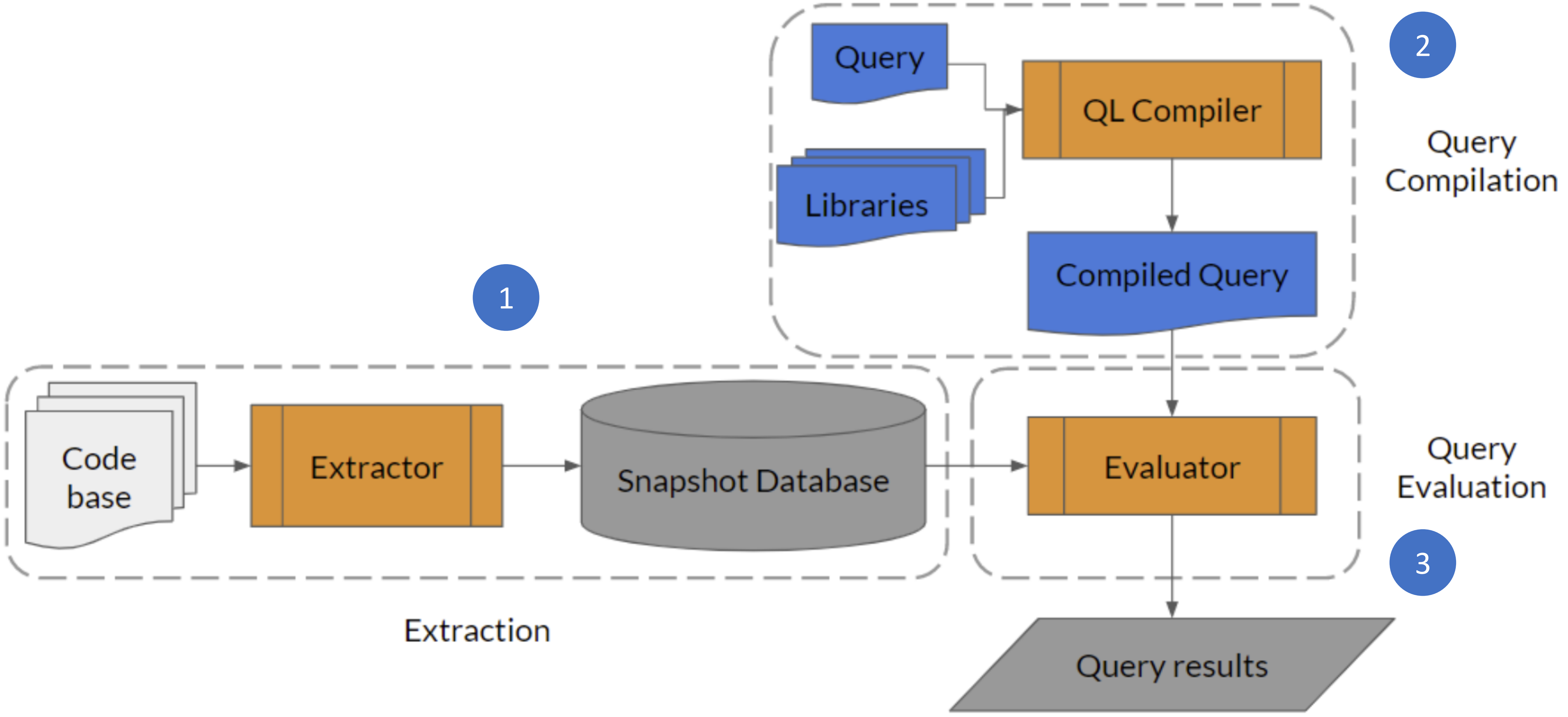
Variant Analysis

- Manual code analysis/audit looks for bugs that lead to security vulnerabilities
- Variant analysis is the process of modeling a known vulnerability and looking for similar issues in the code:
  - Model the security problem so that it can be applied to a representation of the program
- Scan the code base looking for instances of the modeled security issue
- Add the model to a repository and use it in the application build process (continuous integration)
- The CodeQL system is a language and platform for automating variant analysis
  - Find bugs, security vulnerabilities, perform analysis systematically and with the possibility of sharing knowledge

CodeQL

- System developed by GitHub
- The QL (Query Language) language is:
    - A logical language based on first-order logic
    - A declarative language without side-effects
    - An object-oriented language
    - A query language over a read-only CodeQL database
- It is a platform
    - Query Analysis Engine
    - command line tools

CodeQL Architecture



Example: a *bug*

```
int write(int[] buf, int size, int loc, int val) {
  if (loc >= size) {
    // return -1;
  }

  buf[loc] = val;

  return 0;
}
```

- The return instruction has been commented out (e.g. for debugging purposes)
- The if statement is now useless code
- Without explicit bounds checking the code will throw `ArrayIndexOutOfbounds`

Query example about Java code

Reuse of logic about the language under analysis

```
import java
```

```
from IfStmt ifstmt, BlockStmt block
where
  block = ifstmt.getThen() and
  block.getNumStmt() = 0
select ifstmt, "This if-statement is redundant."
```

Interrogation that describes what you are trying to find

- A query file has the extension .ql and contains a query clause and, optionally, predicates, classes and modules.

Predicate

```
import java


from IfStmt ifstmt, BlockStmt block
where
  block = ifstmt.getThen() and
  block.getNumStmt() = 0
select ifstmt,
       "This if-statement is redundant."
```

```
import java

predicate isEmpty(BlockStmt block) {
  block.getNumStmt() = 0
}

from IfStmt ifstmt
where isEmpty(ifstmt.getThen())
select ifstmt
```

- A predicate allows you to highlight part of the interrogation

Classes

- Classes in QL extend one or more types, represent a set of values, and define predicates

```
class OneTwoThree extends int {
  OneTwoThree() { this = 1 or this = 2 or this = 3 } // characteristic predicate

  // member predicate
  string getAString() { result = "One, two or three: " + this.toString() }

  // member predicate
  predicate isEven() {this = 2 }
}
```

- The definition of the class body consists of a feature predicate (optional) and one or more member predicates.

Classes

```
import java

predicate isEmpty(BlockStmt block) {
  block.getNumStmt() = 0
}

from IfStmt ifstmt
where isEmpty(ifstmt.getThen())
select ifstmt
```

```
import java

class EmptyBlock extends BlockStmt {
  EmptyBlock() {
    this.getNumStmt() = 0
  }
}

from IfStmt ifstmt
where ifstmt.getThen() instanceof
      EmptyBlock
select ifstmt
```

- The EmptyBlock class is a BlockStmt whose number of instructions is zero

Successively refine the code

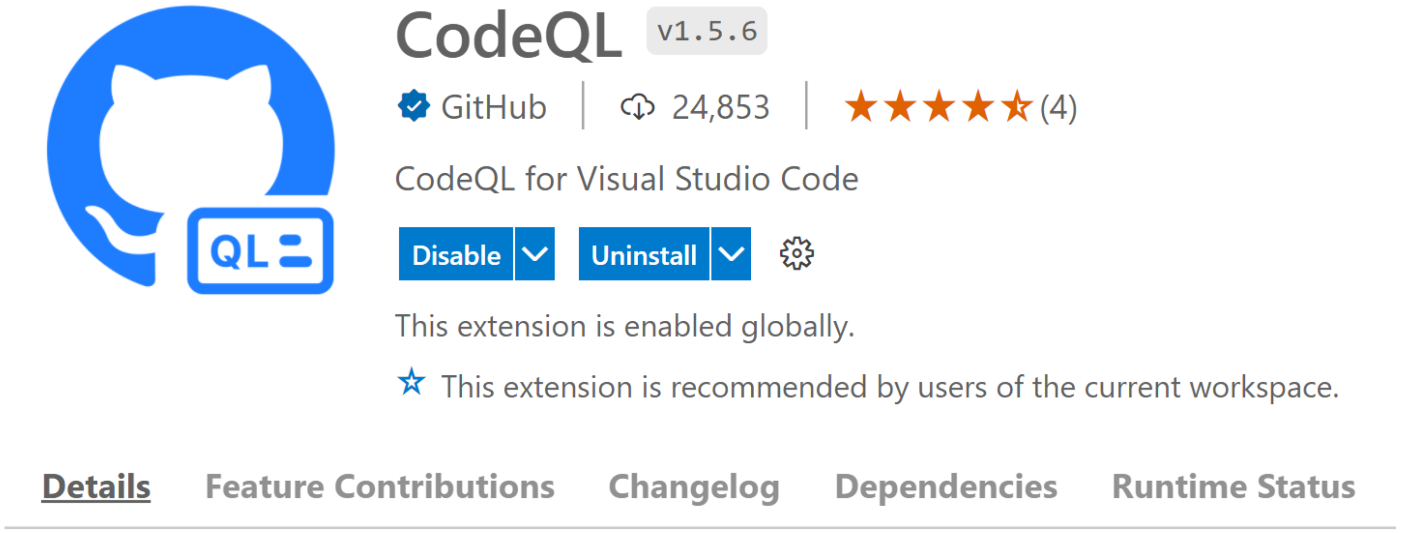
- Looks for if statements with no body in then and no code in else

```
import java

class EmptyBlock extends BlockStmt {
  EmptyBlock() { this.getNumStmt() = 0 }
}

from IfStmt ifstmt
where
  ifstmt.getThen() instanceof EmptyBlock and not exists(ifstmt.getElse())
select ifstmt, "This if-statement is redundant."
```

Installation of CodeQL and extension in VSCode

- Command Line Interface:
  - https://github.com/github/codeql-cli-binaries/releases

- https://codeql.github.com/docs/codeql-for-visual-studio-code/setting-up-codeql-in-visual-studio-code/



CodeQL v1.5.6

GitHub | 24,853 | ★★★★☆ (4)

CodeQL for Visual Studio Code

This extension is enabled globally.

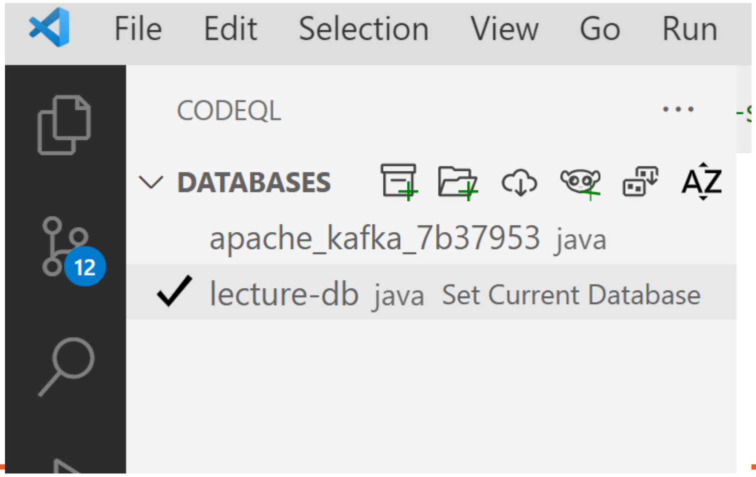
☆ This extension is recommended by users of the current workspace.

Usage

- The installation of the extension includes a database for the Apache Kafka project and queries in some languages
  - https://kafka.apache.org/,
- Creation of new databases

```
$ codeql database create <db path> -l java
```

  - Command assumes source code of project to analyze is in current directory
  - It must be possible to compile the project with automatic build tool (eg Maven, Gradle)
- Database management in the VSCode extension
- Extension allows you to view AST



apache_kafka_7b37953 java
✓ lecture-db java Set Current Database

Flow analysis in Java

- The DataFlow module defines the Node class that represents any element through which it can pass information (expressions, parameters, etc.)
- The TaintTracking module does local flow analysis with tainting

```
TaintTracking::localTaint(DataFlow::parameterNode(source), DataFlow::exprNode(sink))
```

- The Configuration class does global flow analysis
- How to perform a query that detects the creation of HTTP connections from a value obtained from an environment variable?

Interrogation summary

```
/**
 * Finds environment variable used to create an URL object
 */

import java

class GetenvSource extends DataFlow::ExprNode { ... }

class GetenvToURLConfiguration extends DataFlow::Configuration { ... }

from DataFlow::Node src, DataFlow::Node sink, GetenvToURLConfiguration config
where config.hasFlow(src, sink)
select src, "This environment variable constructs a URL $@.", sink, "here"
```

Class used for global analysis between sink and source

```
/**
 * Global taint analysis. Source is “GetenvSource”. Sink is a call to the constructor of
 * class java.net.URL. */
import semmle.code.java.dataflow.DataFlow

class GetenvToURLConfiguration extends DataFlow::Configuration {
    GetenvToURLConfiguration() { this = "GetenvToURLConfiguration“ }

    override predicate isSource(DataFlow::Node source) { source instanceof GetenvSource }

    override predicate isSink(DataFlow::Node sink) {
        exists(Call call |
               sink.asExpr() = call.getArgument(0) and
               call.getCallee().(Constructor).getDeclaringType()
                                              .hasQualifiedName("java.net", "URL")
        )
}   }
```

Classe que representa o método System.getenv()

```
class System {
    public static Map<String,String> getenv()
    //...
}
```

```
import java

class GetenvSource extends DataFlow::ExprNode {
    GetenvSource() {
        exists(Method m | m = this.asExpr().(MethodAccess).getMethod() |
            m.hasName("getenv") and
            m.getDeclaringType() instanceof TypeSystem
        )
    }
}
```

$\exists\, m : m\ is\ a\ method, \ldots$

https://codeql.github.com/docs/ql-language-reference/formulas/

CWE-78 Example

```
class Test {
    public static void main(String[] args) throws Exception{
        // BAD: user input might include special characters such as ampersands
        {
            String latlonCoords = args[1];
            Runtime rt = Runtime.getRuntime();
            Process exec = rt.exec("cmd.exe /C latlon2utm.exe " + latlonCoords);
        }
        // GOOD: use an array of arguments instead of executing a string
        {
            String latlonCoords = args[1];
            Runtime rt = Runtime.getRuntime();
            Process exec = rt.exec(new String[] {
                    "c:\\path\\to\\latlon2utm.exe",
                    latlonCoords });
        }
    }
}
```

https://cwe.mitre.org/data/definitions/78.html

https://github.com/github/codeql/tree/main/java/ql/src/Security/CWE/CWE-078

ISEL
INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

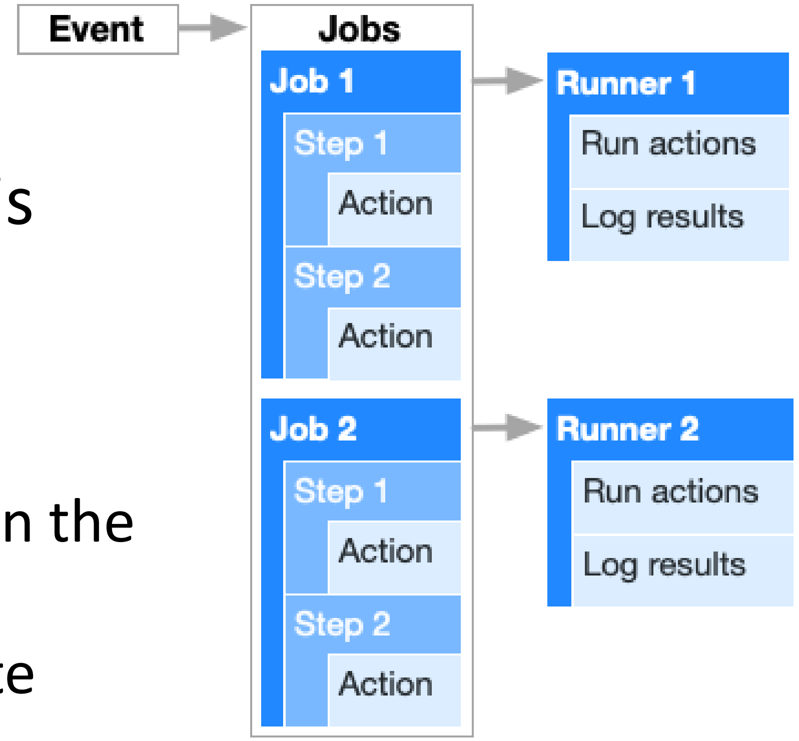
Automating Analysis with GitHub Actions

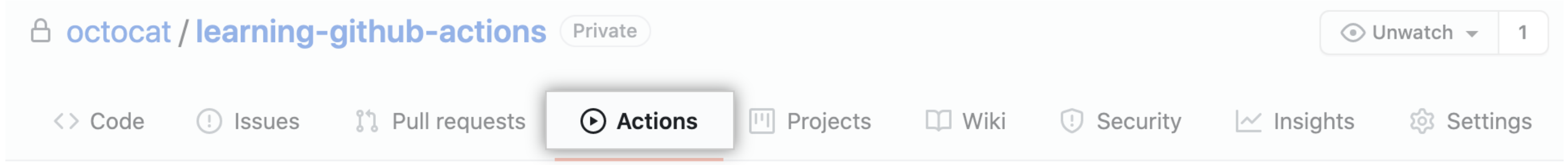
ISEL – Instituto Superior de Engenharia de Lisboa
Rua Conselheiro Emídio Navarro, 1 | 1959-007 Lisboa

Github actions

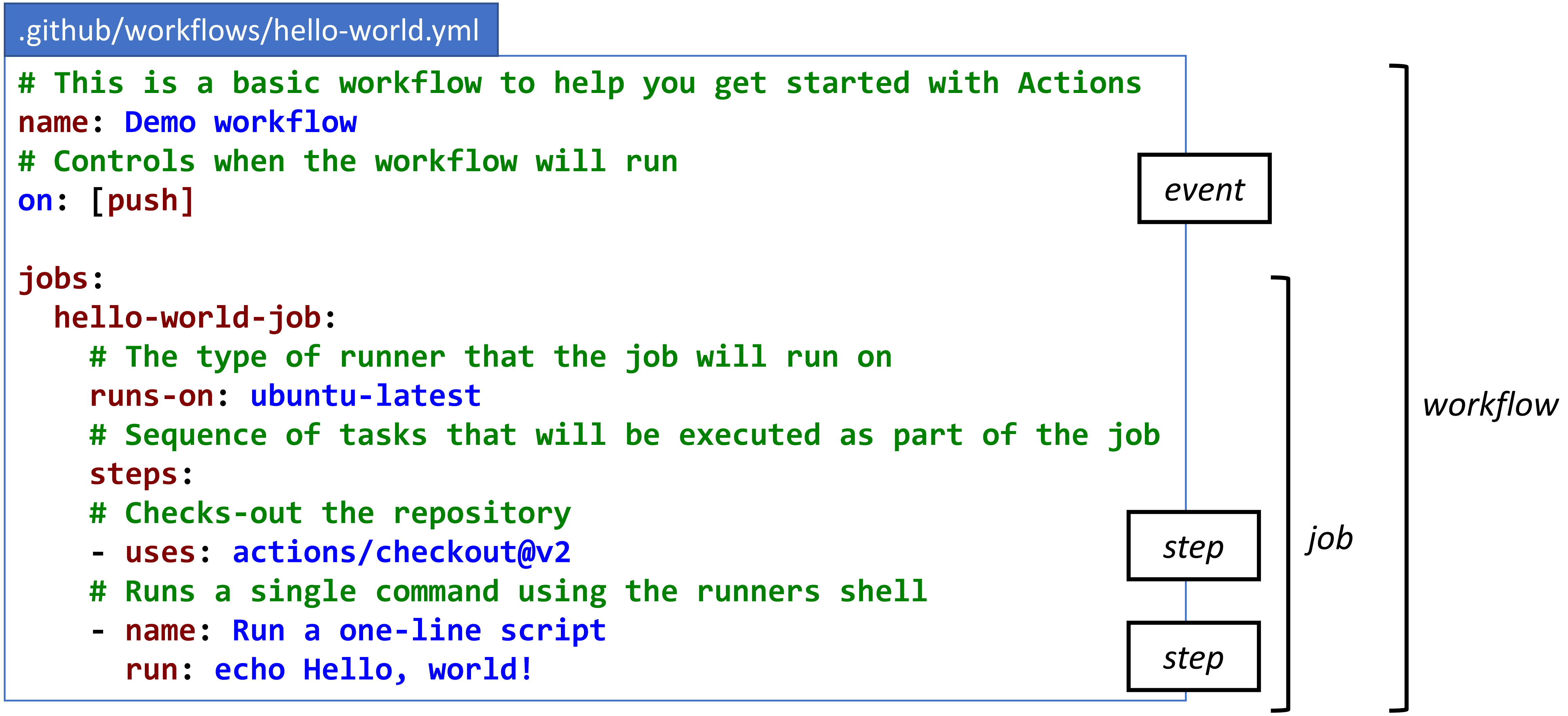
- Operations performed on a repository (push, pull, pull request, …) can trigger several types of actions

- The actions are grouped in a unit called job which is part of a workflow
  - Each job run on servers hosted by GitHub or external (runners)
  - Description of actions is done in text files (.yml) stored in the repository under analysis
  - The status of the shares can be consulted on the website





Github actions – base structure

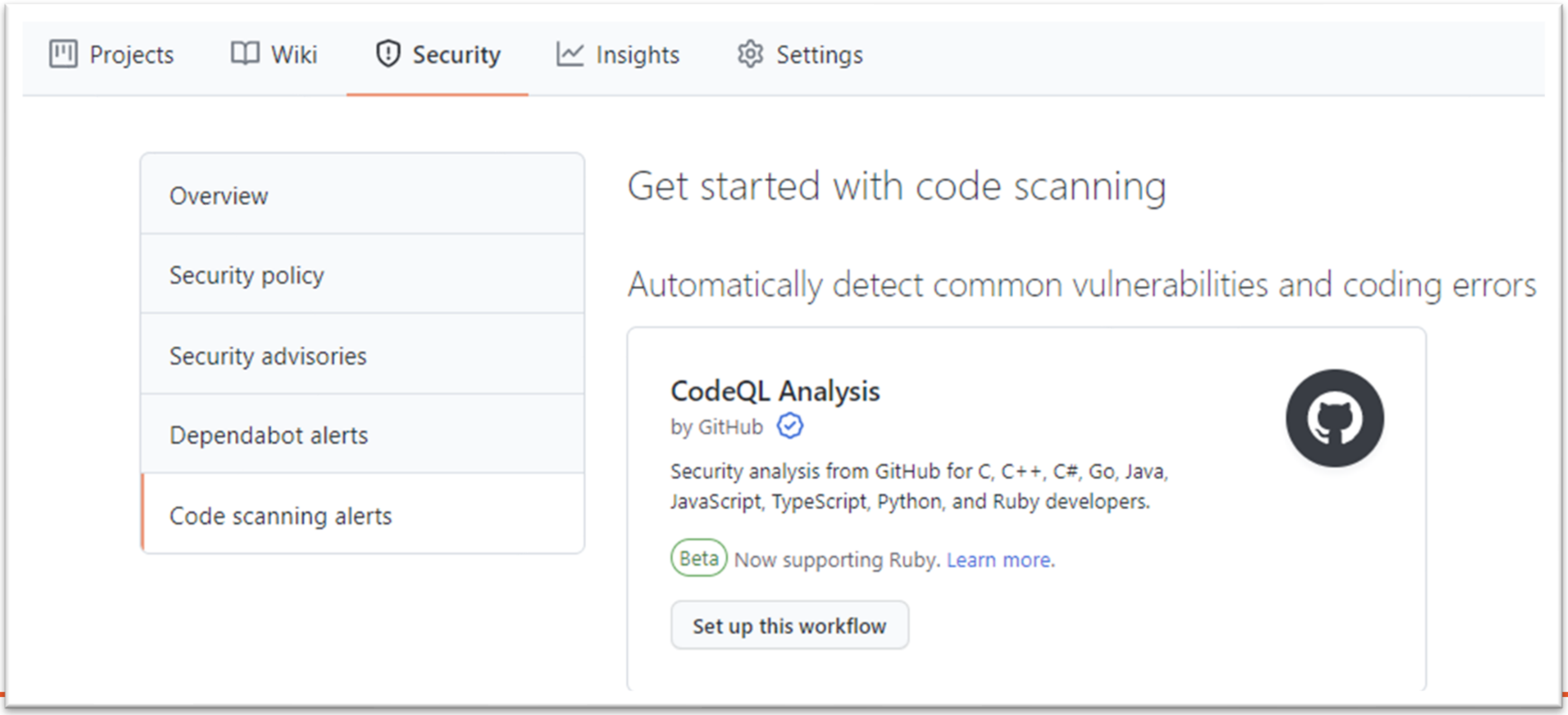


.github/workflows/hello-world.yml

```yaml
# This is a basic workflow to help you get started with Actions
name: Demo workflow
# Controls when the workflow will run
on: [push]

jobs:
  hello-world-job:
    # The type of runner that the job will run on
    runs-on: ubuntu-latest
    # Sequence of tasks that will be executed as part of the job
    steps:
    # Checks-out the repository
    - uses: actions/checkout@v2
    # Runs a single command using the runners shell
    - name: Run a one-line script
      run: echo Hello, world!
```

Github Action for CodeQL

- Coverage of various CWE in Java
- https://codeql.github.com/codeql-query-help/java-cwe/
  - Action source code: https://github.com/github/codeql-action
- In Github: *Security -> Code Scanning Alerts -> Set up this Workflow*



Get started with code scanning

Automatically detect common vulnerabilities and coding errors

CodeQL Analysis

by GitHub

Security analysis from GitHub for C, C++, C#, Go, Java, JavaScript, TypeScript, Python, and Ruby developers.

Beta Now supporting Ruby. Learn more.