Lecture Notes
September 22, 2020

Working in a team: *software reuse*

To reuse previously written code from your own team (same codebase), you can either copy from wherever it is in the codebase and paste into whatever you are working on now, and then modify it to fit (e.g., rename identifiers), or you can call it. In most cases it's best to call it.

Duplicate or near-duplicate code, known as *code clones*, is considered a "code smell". A codebase containing code clones has more code to maintain and is thus more expensive to maintain. It is likely that a bug will be fixed in some copies but not in all copies, and the code may diverge over time making it harder to find the remaining copies of the bug.

Some refactoring tools (e.g., JDeodorant) automatically detect a group of code clones and help developers replace each instance of the clone with a call to a single code unit.

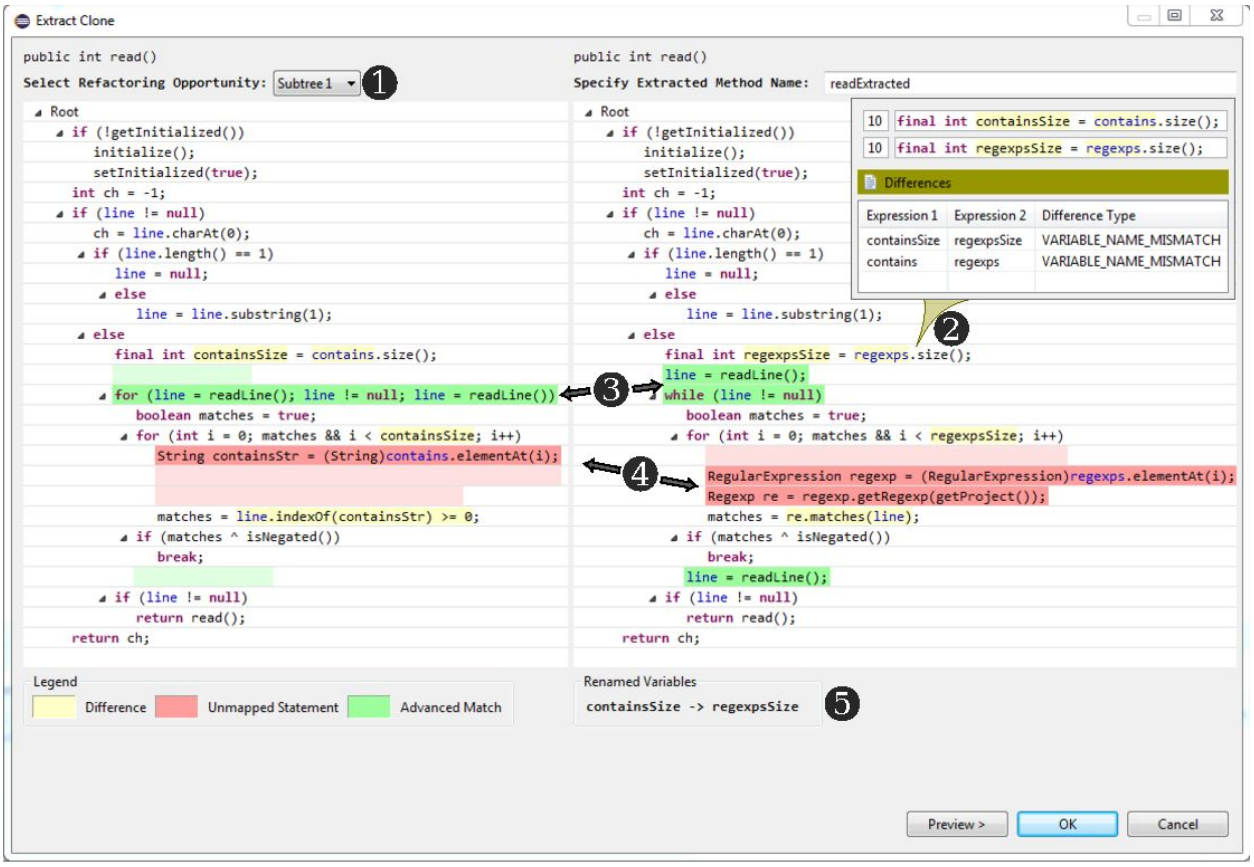


Figure 2: Clone pair visualization and refactorability analysis.

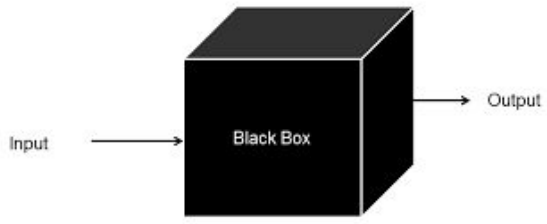
There are similarly two ways to reuse ***third-party code*** (different codebase): 1. Copy/paste that code into your own codebase (and modify it to fit); 2. Leverage the previously written code externally through its API.

What does third-party mean, who are the other two parties? You (or your team) are the first party, your customer or user is the second party, any external code used by your code is considered third-party.

Copy/paste may be the only option since not all open-source code is provided in a form intended for independent reuse, e.g., the code you want to reuse may be part of someone else's unrelated application but happens to 'do the right thing' for your application.

API = application programming interface

Third-party code intended for reuse has an API. Many 'public' APIs are blackbox = You can use the interface provided by the box but you cannot peek inside the box.



Internal behavior of the code is unknown

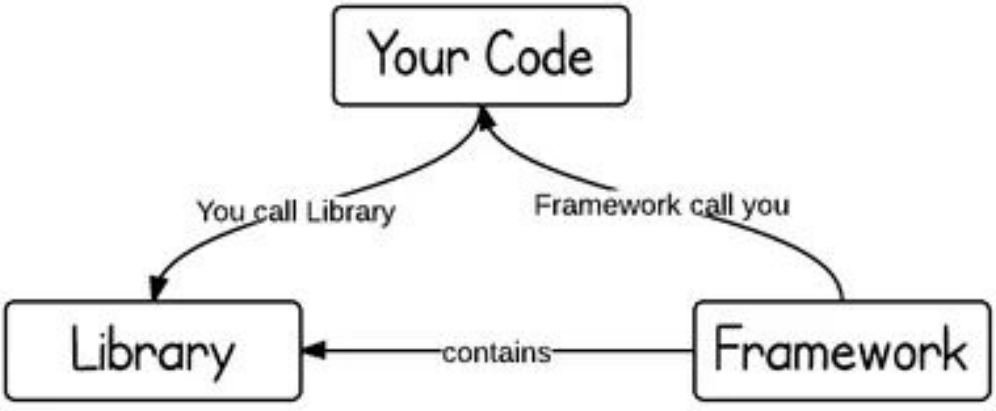
There are three major types of third-party code that are widely used via APIs: frameworks, libraries and services. Frameworks often include libraries and may use services. (APIs can access data, not just code, but right now we consider code.)

Using frameworks, libraries and services (when available) is almost always better, faster, cheaper than developing your application from scratch.

What is the difference between a library and a service? A library is shared code that you compile or bundle into your application. A service is a shared capability that you access from your application.

See Libraries vs. Services for further explanation.

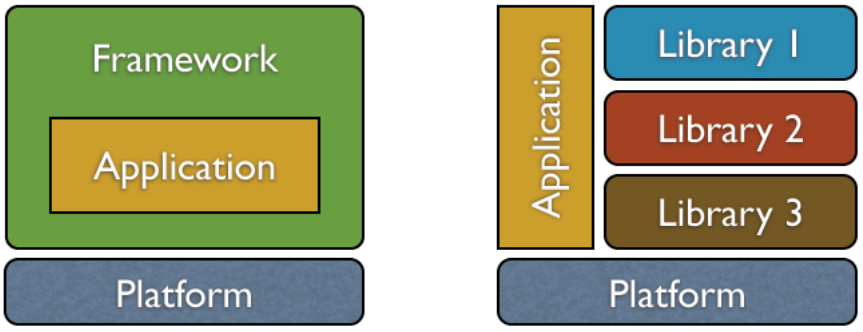
What is the difference between a *framework* and a *library*?



"*Inversion of control*"
(Hollywood Principle = don't call us, we'll call you)

- When your code calls a library (or a service), your code is in control.

- In contrast, the framework calls your code, so the framework is in control. The framework plays the role of the main program in coordinating and sequencing application activity and you fill in the blanks.



A library or service provides a set of functions that can be called from your code, that set of functions is its API.

Most programming languages have standard libraries that “come with” the platform (runtime environment or operating system), e.g., for I/O, string processing, math. Other libraries are downloaded manually or by a build tool or package manager (e.g., Maven central, npm registry) to bundle with your program.

Frameworks provide a set of generic components (some of which may be third-party libraries) that work together to accomplish common development tasks and functionalities for a popular target domain, such as web apps (e.g., https://javalin.io/) or mobile apps (e.g., React Native).

A framework defines an application skeleton with places to put your code for the framework to call. The context of those places comprises part of its API, but there are often also library functions to call.

Some services are very specific to a particular task or domain (e.g., https://ipinfo.io/ maps an IP address to geographic location, carrier, etc.). Some services interface to a particular application or ecosystem (e.g., Twitter API).

Remote access typically uses a ‘REST API’, sometimes called a ‘RESTful API’.

A REST API is usually accessed via HTTP (HyperText Transfer Protocol) or HTTPS (Secure HTTP or HTTP with Security) over the Internet with a predefined set of URLs (Uniform Resource Locators).

A REST API consists of

HTTP METHOD endpoint1 { optional body }
...
HTTP METHOD endpointN { optional body }

for one or more endpoints as the rest of the HTTP ‘header’. The endpoints (routes) are paths to resources. The { optional body } contains additional data.

There are several HTTP methods (operations): GET, HEAD, POST, PUT, PATCH, DELETE, CONNECT, OPTIONS, TRACE. Usually use only GET and POST.

http://example.com/path/to/resource

If you click a URL like this one on a page in your browser, the browser will use HTTP to connect to the server at example.com (domain) and access /path/to/resource

This typically fetches some content located at /path/to/resource, which is then returned to and rendered by your browser.

Or this URL might run a program located at /path/to/resource and then return the results of running that program to be rendered by your browser.

In either case, the ‘rendering’ could give you text, image, audio, video, code, or any other data.

Called REST = REpresentational State Transfer because accessing the resource *transfers* to the caller a *representation* of the current *state* of the resource.

The HTTP caller (requester) can be any program, it does not need to be a web browser. And the response does not have to come from a conventional web server. When the HTTP caller is your program, you are using a REST API.

Addendum:

What is a ‘representation’?

The same resource (or resource state) may have many different representations (variants).

A page of text might be presented in HTML or plain text.

The words in the text might be written in English or Chinese.

The characters in the text might be encoded in ASCII or UTF-8.

The page might be organized differently for a PC versus a mobile device.

The requester might specify the representation(s) it can handle by including optional fields in the HTTP header.

Or the responder might send multiple representations and the requester uses those it can handle.

Determining what to send/receive is called *content negotiation*. More info here.

http://example.com/path/to/resource?querystring
(a querystring is something like
field1=value1&field2=value2&field3=value3)

If you click a query URL like this one in your browser, the browser will use HTTP to connect to the server at example.com, tell the program at the /path/to/resource endpoint to run, and give it the parameters value1, value2 and value3. The browser then renders the results of running that program with the parameters.

Again, accessing the resource *transfers* to the caller a *representation* of the *state* of the resource.

Typically a query URL would not be sitting there embedded in a page with the field values already filled in. Instead the values of the fields would likely be computed somehow by the browser, possibly by showing the user a form to fill in. The form displays the fields and the user enters the values.

Again the HTTP caller can be any program, it does not need to be a web browser. When the caller is your program, you are again using a REST API - your program computes the values for the fields and the recipient server responds with the results for those parameters

Where are the GET and POST?

A URL like this one

http://example.com/path/to/resource

should not change the state of the resource.

If your API call only intends to fetch something with no side-effects, use GET.

```
app.get("/path/to/resource", ctx -> {
    // code to fetch or construct web page
    ctx.html(html); // return page to display
});
```

A URL query on http://example.com/path/to/resource with field1=value1&field2=value2&field3=value3

should not have side-effects.

If your API call only intends to fetch something, e.g., the field values will be used for a search, then use GET with the query fields/values in the HTTP header.

```
app.get("/path/to/resource", ctx -> {
  // some code
  // do something with ctx.queryParam(field1)
  // do something with ctx.queryParamMap()
  // some code
  // construct json object
  ctx.json(object); // return data
});
```

Using http://example.com/path/to/resource in the header and field1=value1&field2=value2&field3=value3 in body

If your API call intends to *change* the state of the resource based on the provided field (form) values, then use POST and put the fields/values in the body of the HTTP request rather than the header.

```
app.post("/path/to/resource", ctx -> {
  // some code
  // do something with ctx.formParam(field1)
  // do something with ctx.formParamMap()
  // some code
  ctx.status(200); // http OK
});
```

Many short examples at https://javalin.io/documentation.

You need to use a REST API for the individual project. Here is more written information about REST APIs in general, and here is a video series about using REST APIs with Javalin (much more than you need to know!).

Addendum: Another short REST API description here.

You will need to use *some* API, not necessarily REST, in your team project. You do not need to use a framework - but your project may go smoother if you do.

“Assignment I1”: https://courseworks2.columbia.edu/courses/104335/assignments/482592

The next individual programming assignment builds on this one, using JUnit and SpotBugs to find bugs in your tic-tac-toe code. There will be an in-class tutorial corresponding to this assignment on Thursday.

“Assignment T0”: https://courseworks2.columbia.edu/courses/104335/assignments/486855 This is the initial team assignment, to form a team and choose which programming language your team will use (C/C++, Java, Javascript, Python).

It’s not due until October 15, after the individual project and first assessment, but you can start forming teams any time.

There is a “search for teammates” thread in piazza that you can use to look for teammates https://courseworks2.columbia.edu/courses/104335/external_tools/1456

If time permits, at the end of each of the next few class sessions I will put everyone in breakout groups so you can introduce yourself and meet each other. Today (or next class if no time today) there is a short exercise to work on in your breakout groups, and then volunteers will report their group’s ideas to the class:

Recall the small drone that flies around an in-person classroom, just below the ceiling, and uses its sensors to determine which students are “paying attention”.

We previously considered how to do acceptance testing. Today, consider how to do ***beta testing*** with live users in real (college-level) classrooms with real students and a real lecture in progress.

- The drone software should make best effort to avoid reporting ***false negatives*** = fails to report a student who is not paying attention.
- And make best effort to avoid reporting ***false positives*** = reports student as not paying attention when actually she is paying attention.
- What will you use for the test oracle?

Ideally, testing has a “test oracle” to determine whether the results of each test is right or wrong. The test oracle may be automated (e.g., assertions) or human judgment. Finding a test oracle may be hard, or impossible, for some software or parts of some software. There will be a lot of discussion about test oracles later in the course.

Beta testing is different from acceptance testing. When representatives of the user organization participate in acceptance testing, their goal is to test the software to check that it is acceptable - it fulfills their needs. During beta testing, users have agreed to try out the software while they go about their normal activities. Their main goal is to accomplish their activities, not to test.

For your team project proposals, you will need to devise acceptance tests. We do not expect you to recruit users to beta test, but if you do that's great!

Please ***return to the main room*** to report after the breakout rooms close (after 10 minutes if they don't close).