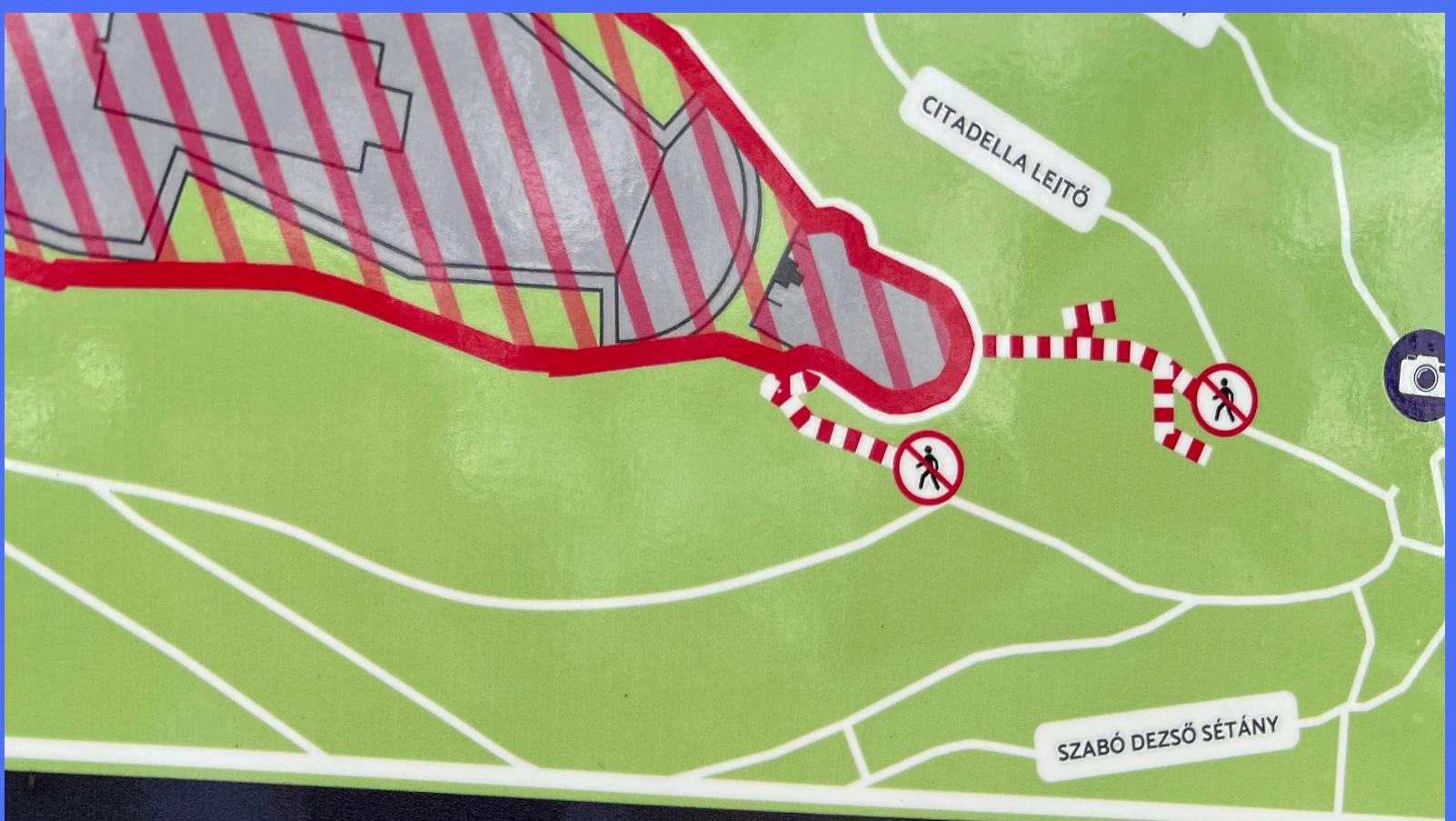


Lars Eckart, Technical Coach, Co-Maintainer ApprovalTests.Java

# Approval Tests



How I learned about Approval Tests  
and why I want you to know about it too

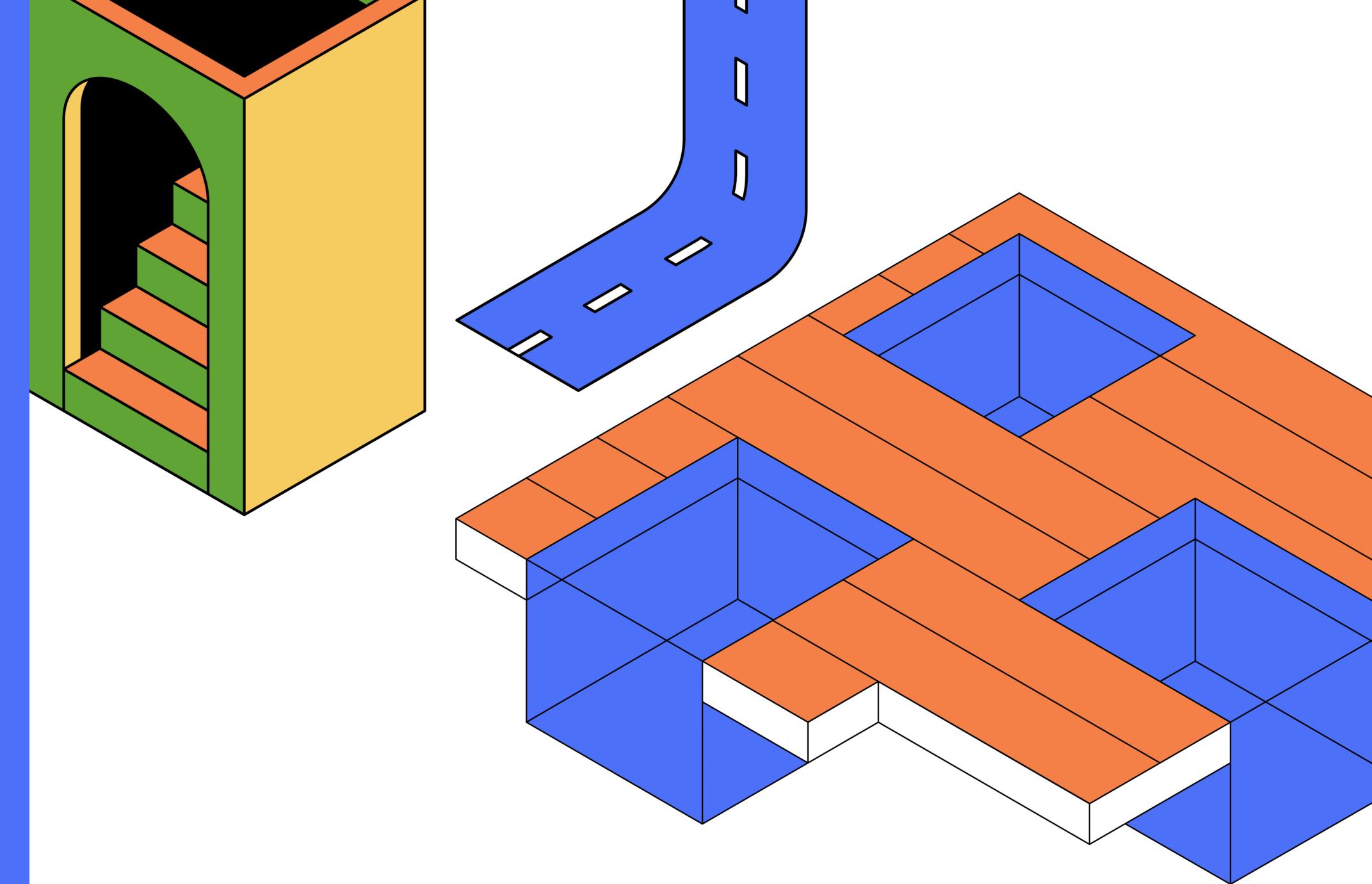


## DEAR VISITOR!

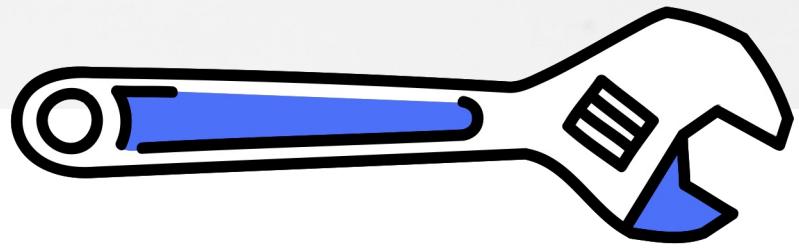
After decades of neglect, we have started to refurbish the Citadella and its surroundings. The work involves restoration of the complete fort building and the laying out of new green spaces. There will also be a new public park for visitors to relax in. In the Cannon Tower, an exhibition commemorating the Hungarian people's struggles for freedom will be opened.

The works require us to close off the roads leading up to the fort, the Freedom Statue and the panorama terrace. The car park at the Citadella will also be closed during the refurbishment. While the work is in progress, viewpoints on the surrounding streets – Citadella lejtő, Szabadság sétány and Szabó Dezső sétány – will be open, offering fine views of Budapest, as will the lookout point at the end of Citadella sétány, with a view towards the Buda Castle District.

THANK YOU FOR YOUR PATIENCE AND UNDERSTANDING.

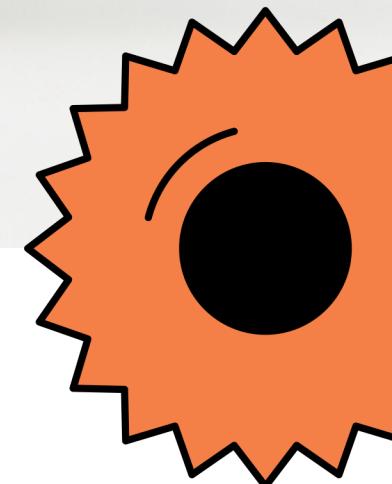


from practice  
to production



Regular Unit Test

Specific Name  
Arrange Act Assert  
`assertEquals(expected, actual)`



```
class GildedRoseTest {  
    @Test  
    void aged_brie_increases_in_quality_the_older_it_gets() {  
        Item[] items = new Item[]{new Item(name: "Aged Brie", sellIn: 7, quality: 10);  
        GildedRose app = new GildedRose(items);  
  
        app.updateQuality();  
  
        assertEquals(11, items[0].quality);  
    }  
}
```

```
assertEquals(11, · items[0].quality);
```



# Approval Test

Name  
Arrange Act Assert  
Verify(actual)

```
@Test  
void approval() {  
    Item[] items = new Item[]{new Item( name: "Aged Brie", sellIn: 7, quality: 10});  
    GildedRose app = new GildedRose(items);  
  
    app.updateQuality();  
  
    Approvals.verify(items[0]);  
}
```

### ☰ GildedRoseTest.approval.approved.txt ×

```
1 Aged Brie, 6, 11|
```

```
Approvals.verify(items[0]);
```



## Characterization

Michael Feathers

## Pinning

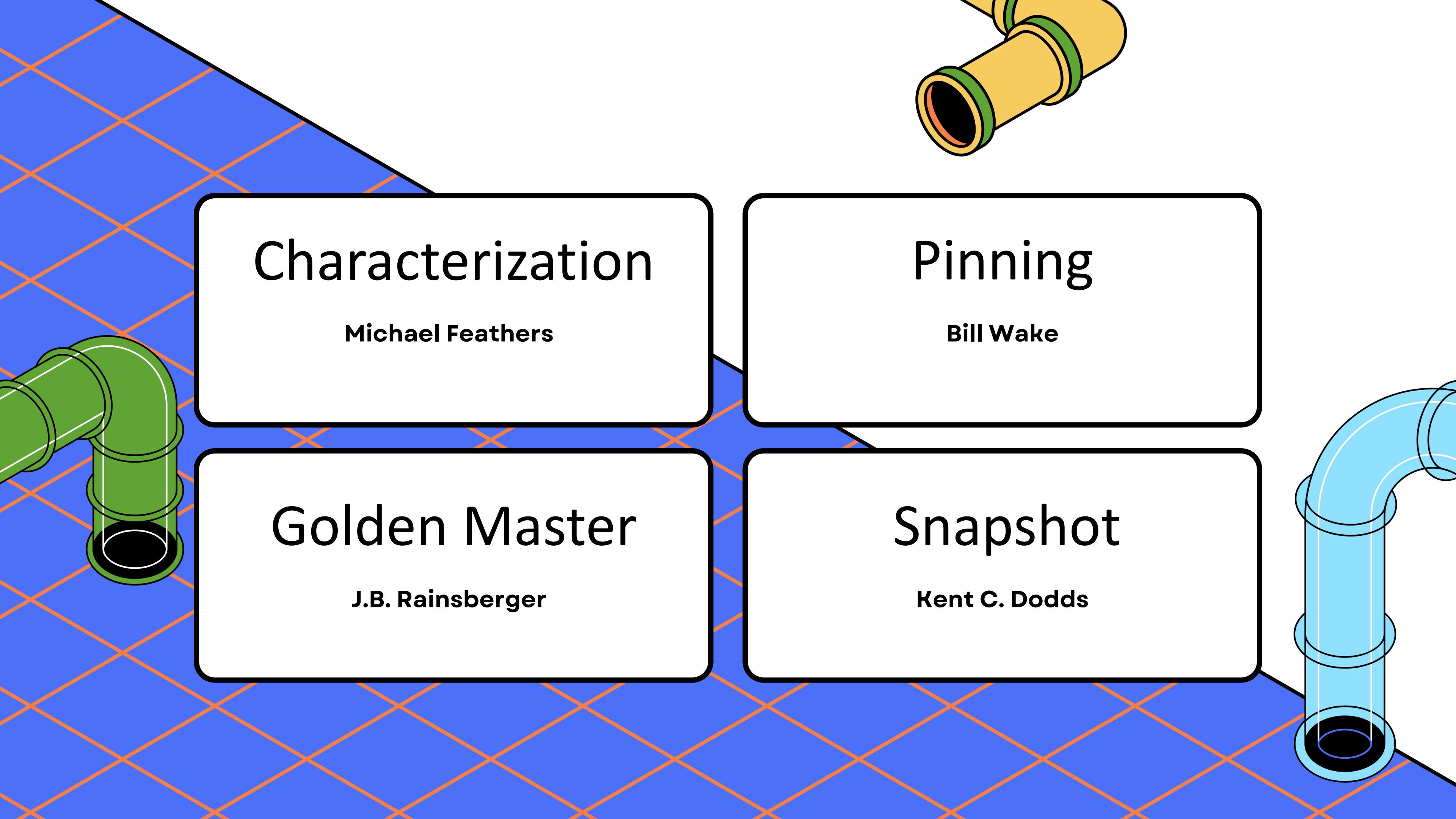
Bill Wake

## Golden Master

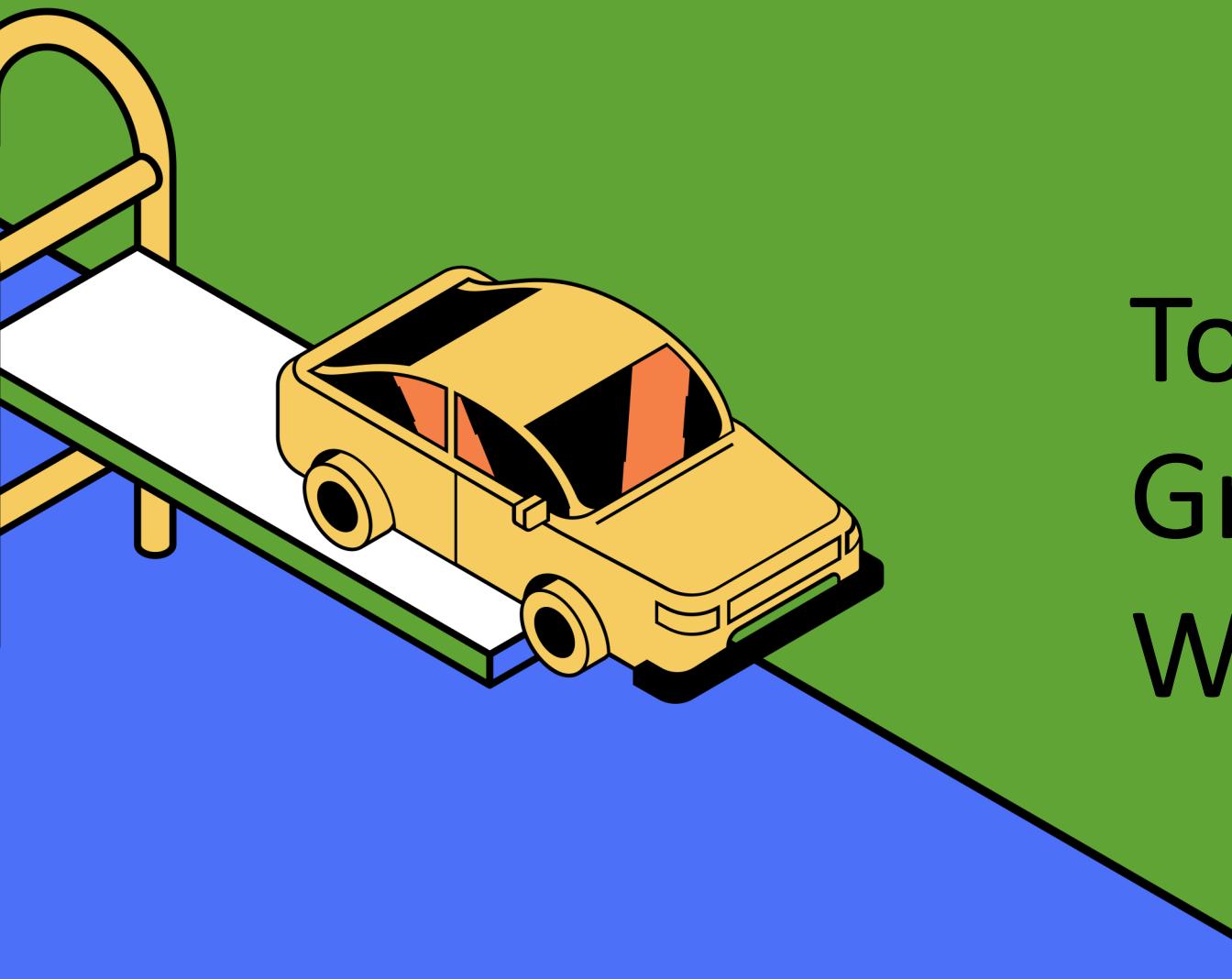
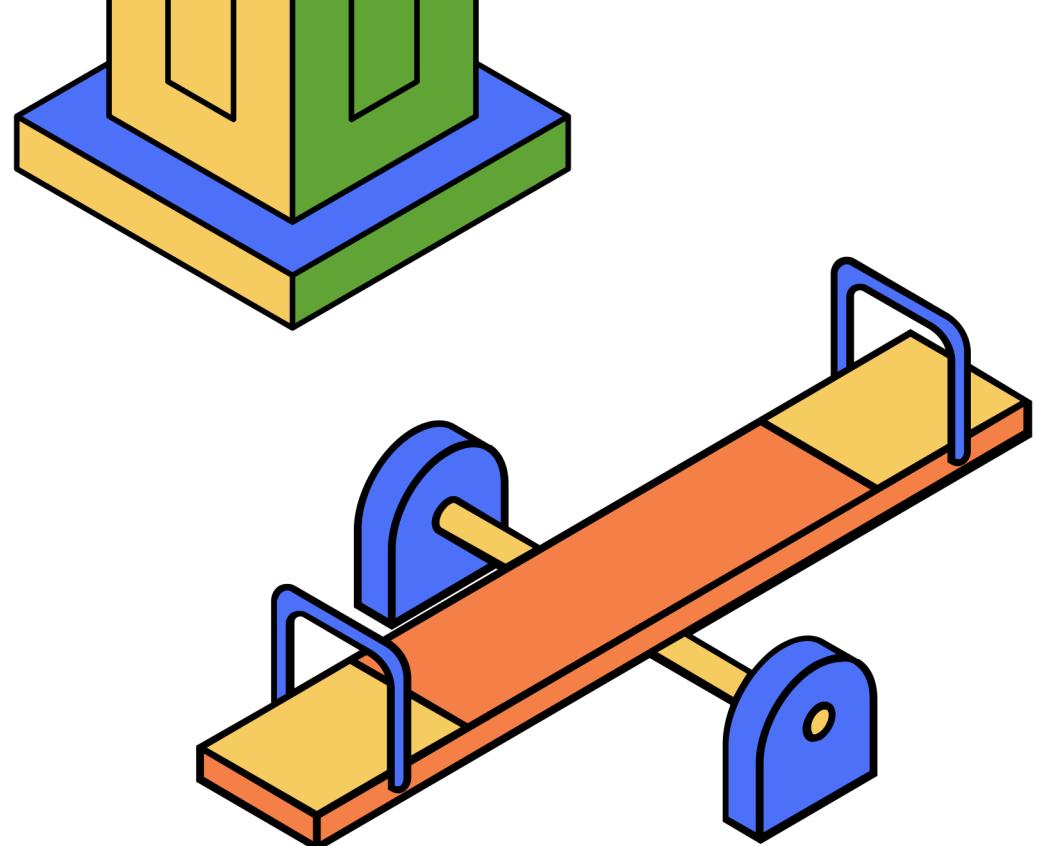
J.B. Rainsberger

## Snapshot

Kent C. Dodds



<https://approvaltests.com>



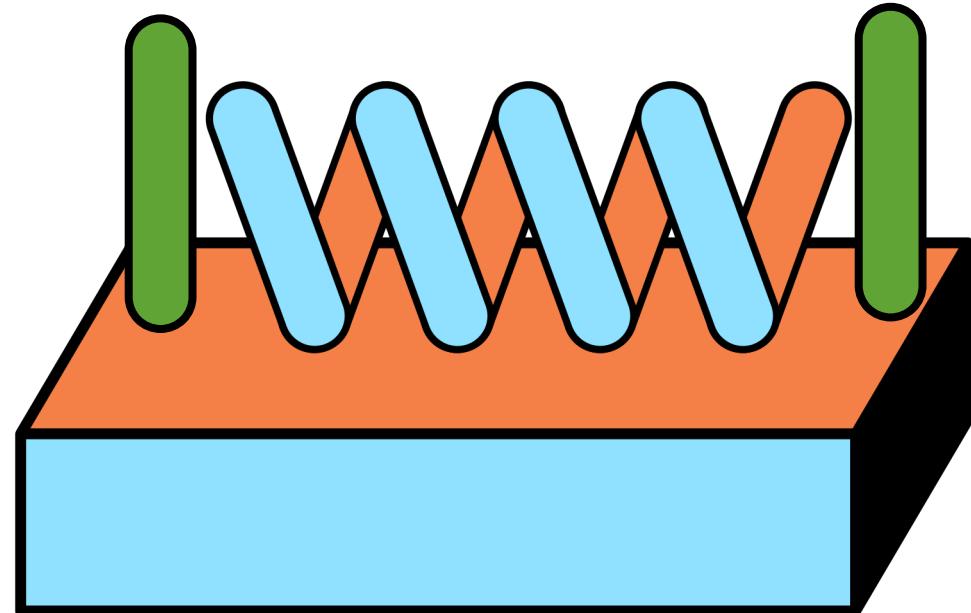
Tomorrow, 14 o'clock,  
Green Tent  
Writing Documentation



Llewellyn Falco

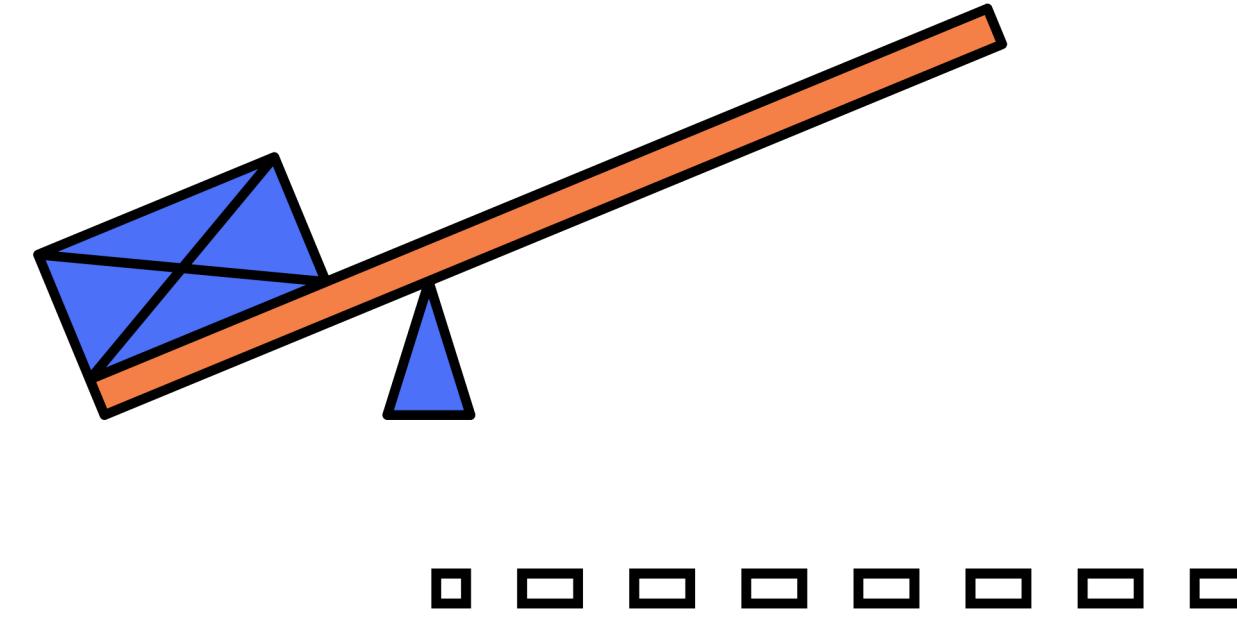
`Approvals.verify`

**String, Object, Map,  
ResultSet, File**



`Approvals.verifyAll`

**Arrays, Iterable**



`JsonApprovals`

`AwtApprovals`

**String, Image,  
Component, Paintable**

`CombinationApprovals`  
`verifyAllCombinations`

# Acceptance Tests

01

Keycloak CLI

```
1  Keycloak - Open Source Identity and Access Management
2
3  Find more information at: https://www.keycloak.org/docs/latest
4
5  Usage:
6
7  kc.bat [OPTIONS] [COMMAND]
8
9  Use this command-line tool to manage your Keycloak cluster.
10 Make sure the command is available on your "PATH" or prefix it with "./" (e.g.:
11 "./kc.bat") to execute from the current folder.
12
13 Options:
14
15 -cf, --config-file <file>
16           Set the path to a configuration file. By default, configuration properties are
17           read from the "keycloak.conf" file in the "conf" directory.
18 -h, --help
19           This help message.
20 -v, --verbose
21           Print out error details when running this command.
22 -V, --version
23           Show version information
24
25 Commands:
26 build      Creates a new and optimized server image.
27 start      Start the server.
28 start-dev   Start the server in development mode.
29 export     Export data from realms to a file or directory.
30 import     Import data from a directory or a file.
31 show-config
32 tools      Utilities for use and interaction with the server.
33 completion Generate bash/zsh completion script for kc.bat.
34
35 Examples:
36
37 Start the server in development mode for local development or testing:
38
39 $ kc.bat start-dev
40
41 Building an optimized server runtime:
42
43 $ kc.bat build <OPTIONS>
```



02

Betslip sharing

03

MicroService API

```
@Test
void storeAndRetrieveMovies() throws JsonProcessingException {
    webTestClient.post() RequestBodyUriSpec
        .uri( uri: "/api/movies" ) RequestBodySpec
        .contentType(MediaType.APPLICATION_JSON)
        .bodyValue("{\"title\": \"The Super Mario Bros. Movie\", \"year\": 2023}")
        .exchange();

    webTestClient.post() RequestBodyUriSpec
        .uri( uri: "/api/movies" ) RequestBodySpec
        .contentType(MediaType.APPLICATION_JSON)
        .bodyValue("{\"title\": \"The Matrix\", \"year\": 1999}")
        .exchange();

    String response = webTestClient.get() RequestHeadersUriSpec<capture of?>
        .uri( uri: "/api/movies" ) capture of?
        .exchange() ResponseSpec
        .returnResult(String.class).getResponseBody().blockFirst();

    Object json = mapper.readValue(response, Object.class);
    String indented = mapper.writerWithDefaultPrettyPrinter().writeValueAsString(json);

    Approvals.verify(indented);
}
```

```
@Test
@Launch({ "--help" })
void testHelp(LaunchResult result) {
    CLIResult cliResult = (CLIResult) result;
    assertHelp(cliResult);
}

@Test
@Launch({ "-h" })
void testHelpShort(LaunchResult result) {
    CLIResult cliResult = (CLIResult) result;
    assertHelp(cliResult);
}
```

```
1  Keycloak – Open Source Identity and Access Management
2
3  Find more information at: https://www.keycloak.org/docs/latest
4
5  Usage:
6
7  kc.bat [OPTIONS] [COMMAND]
8
9  Use this command-line tool to manage your Keycloak cluster.
10 Make sure the command is available on your "PATH" or prefix it with "./" (e.g.:
11 "./kc.bat") to execute from the current folder.
12
13 Options:
14
15 -cf, --config-file <file>
16                                     Set the path to a configuration file. By default, configuration property
17                                     read from the "keycloak.conf" file in the "conf" directory.
18 -h, --help
19                                     This help message.
20 -v, --verbose
21                                     Print out error details when running this command.
22                                     Show version information
23
24 Commands:
25
26 build
27                                     Creates a new and optimized server image.
28 start
29                                     Start the server.
30 start-dev
31                                     Start the server in development mode.
32 export
33                                     Export data from realms to a file or directory.
34 import
35                                     Import data from a directory or a file.
36 show-config
37                                     Print out the current configuration.
38 tools
39                                     Utilities for use and interaction with the server.
40 completion
41                                     Generate bash/zsh completion script for kc.bat.
42
43 Examples:
44
45 Start the server in development mode for local development or testing:
46
47 $ kc.bat start-dev
48
```

# Use cases from production

01

## Keycloak CLI

```
1  Keycloak - Open Source Identity and Access Management
2
3  Find more information at: https://www.keycloak.org/docs/latest
4
5  Usage:
6
7  kc.bat [OPTIONS] [COMMAND]
8
9  Use this command-line tool to manage your Keycloak cluster.
10 Make sure the command is available on your "PATH" or prefix it with "./" (e.g.:
11 "./kc.bat") to execute from the current folder.
12
13 Options:
14
15 -cf, --config-file <file>
16           Set the path to a configuration file. By default, configuration properties are
17           read from the "keycloak.conf" file in the "conf" directory.
18 -h, --help
19           This help message.
20 -v, --verbose
21           Print out error details when running this command.
22 -V, --version
23           Show version information
24
25 Commands:
26 build      Creates a new and optimized server image.
27 start      Start the server.
28 start-dev   Start the server in development mode.
29 export     Export data from realms to a file or directory.
30 import     Import data from a directory or a file.
31 show-config
32 tools      Utilities for use and interaction with the server.
33 completion Generate bash/zsh completion script for kc.bat.
34
35 Examples:
36
37 Start the server in development mode for local development or testing:
38
39 $ kc.bat start-dev
40
41 Building an optimized server runtime:
42
43 $ kc.bat build <OPTIONS>
```



02

## Betslip sharing

03

## MicroService API

```
@Test
void storeAndRetrieveMovies() throws JsonProcessingException {
    webTestClient.post() RequestBodyUriSpec
        .uri( uri: "/api/movies" ) RequestBodySpec
        .contentType(MediaType.APPLICATION_JSON)
        .bodyValue("{\"title\": \"The Super Mario Bros. Movie\", \"year\": 2023}")
        .exchange();

    webTestClient.post() RequestBodyUriSpec
        .uri( uri: "/api/movies" ) RequestBodySpec
        .contentType(MediaType.APPLICATION_JSON)
        .bodyValue("{\"title\": \"The Matrix\", \"year\": 1999}")
        .exchange();

    String response = webTestClient.get() RequestHeadersUriSpec<capture of?>
        .uri( uri: "/api/movies" ) capture of?
        .exchange() ResponseSpec
        .returnResult(String.class).getResponseBody().blockFirst();

    Object json = mapper.readValue(response, Object.class);
    String indented = mapper.writerWithDefaultPrettyPrinter().writeValueAsString(json);

    Approvals.verify(indented);
}
```

```

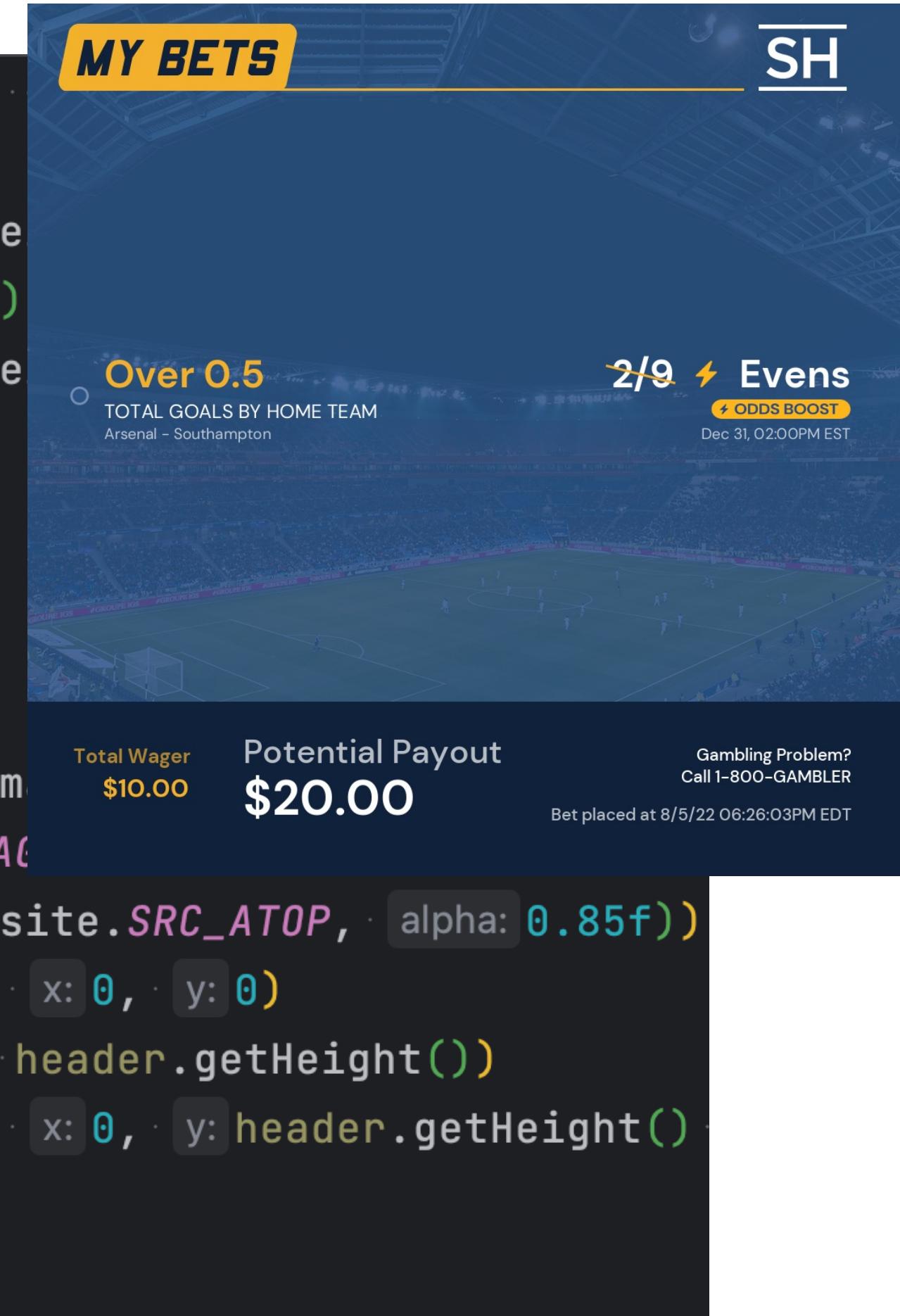
public BufferedImage buildImage(BetSlipImageGenerationRequest request)

    BufferedImage header = buildHeader(request, getWidth(), getHeaderHeight());
    BufferedImage body = buildBody(request, getWidth(), getBodyHeight());
    BufferedImage footer = buildFooter(request, getWidth(), getFooterHeight());

    Color bgColor = Platform.SUGARHOUSE.equals(request.platform())
        ? COLOR_SUGAR_HOUSE_BG
        : COLOR_BET_RIVERS_BG;

    return new BufferedImageBuilder(getWidth(), getHeight(), BufferedImage.TYPE_INT_ARGB)
        .addImage(new ImageElement(getClass().getResource(name: IMAGE_NAME)))
        .addOverlay(bgColor, AlphaComposite.getInstance(AlphaComposite.SRC_ATOP, alpha: 0.85f))
        .addSection(header, header.getWidth(), header.getHeight(), x: 0, y: 0)
        .addSection(body, body.getWidth(), body.getHeight(), x: 0, header.getHeight())
        .addSection(footer, footer.getWidth(), footer.getHeight(), x: 0, y: header.getHeight())
        .build();
}

```

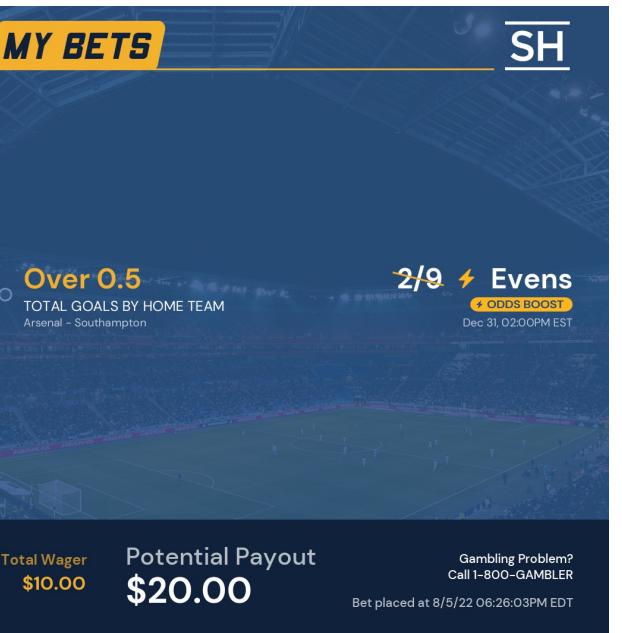


# Use cases from production

01

## Keycloak CLI

```
1  Keycloak - Open Source Identity and Access Management
2
3  Find more information at: https://www.keycloak.org/docs/latest
4
5  Usage:
6
7  kc.bat [OPTIONS] [COMMAND]
8
9  Use this command-line tool to manage your Keycloak cluster.
10 Make sure the command is available on your "PATH" or prefix it with "./" (e.g.:
11 "./kc.bat") to execute from the current folder.
12
13 Options:
14
15 -cf, --config-file <file>
16           Set the path to a configuration file. By default, configuration properties are
17           read from the "keycloak.conf" file in the "conf" directory.
18 -h, --help
19           This help message.
20 -v, --verbose
21           Print out error details when running this command.
22 -V, --version
23           Show version information
24
25 Commands:
26 build      Creates a new and optimized server image.
27 start      Start the server.
28 start-dev   Start the server in development mode.
29 export     Export data from realms to a file or directory.
30 import     Import data from a directory or a file.
31 show-config
32 tools      Utilities for use and interaction with the server.
33 completion Generate bash/zsh completion script for kc.bat.
34
35 Examples:
36
37 Start the server in development mode for local development or testing:
38
39 $ kc.bat start-dev
40
41 Building an optimized server runtime:
42
43 $ kc.bat build <OPTIONS>
```



02

## Betslip sharing

03

## MicroService API

```
@Test
void storeAndRetrieveMovies() throws JsonProcessingException {
    webTestClient.post() RequestBodyUriSpec
        .uri( uri: "/api/movies" ) RequestBodySpec
        .contentType(MediaType.APPLICATION_JSON)
        .bodyValue("{\"title\": \"The Super Mario Bros. Movie\", \"year\": 2023}")
        .exchange();

    webTestClient.post() RequestBodyUriSpec
        .uri( uri: "/api/movies" ) RequestBodySpec
        .contentType(MediaType.APPLICATION_JSON)
        .bodyValue("{\"title\": \"The Matrix\", \"year\": 1999}")
        .exchange();

    String response = webTestClient.get() RequestHeadersUriSpec<capture of?>
        .uri( uri: "/api/movies" ) capture of?
        .exchange() ResponseSpec
        .returnResult(String.class).getResponseBody().blockFirst();

    Object json = mapper.readValue(response, Object.class);
    String indented = mapper.writerWithDefaultPrettyPrinter().writeValueAsString(json);

    Approvals.verify(indented);
}
```

```
@Test
void storeAndRetrieveMovies() throws JsonProcessingException {
    storeMovie( body: "{\"title\": \"The Super Mario Bros. Movie\"}
    storeMovie( body: "{\"title\": \"The Matrix\", \"year\": 1999}

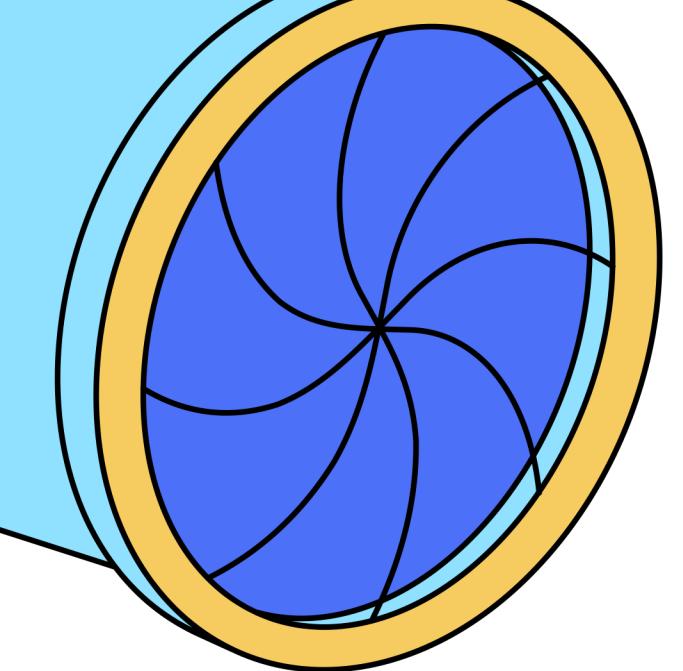
    String response = webTestClient.get() RequestHeadersUriSpec<cap>
        .uri( uri: "/api/movies" ) capture of ?
        .exchange() ResponseSpec
        .returnResult(String.class).getResponseBody().block()

    Object json = mapper.readValue(response, Object.class);
    String indented = mapper.writerWithDefaultPrettyPrinter().wr

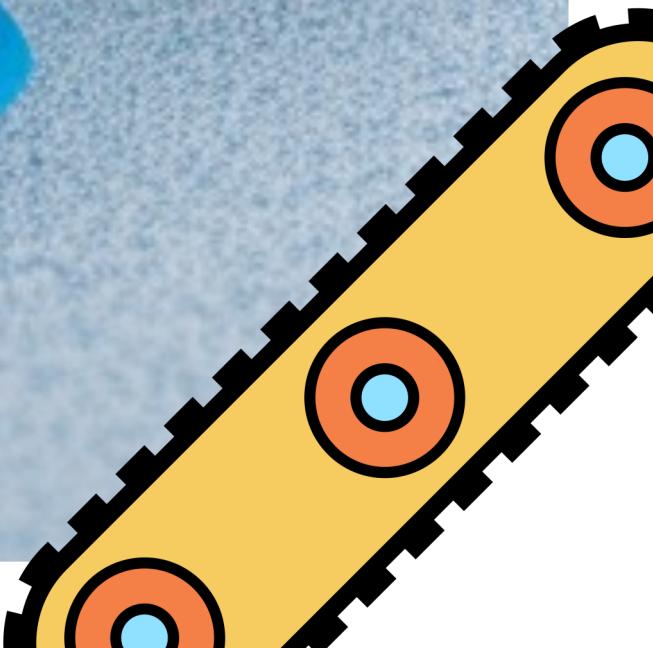
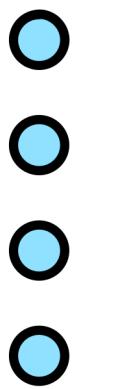
    Approvals.verify(indented);
}
```

### ≡ MovieAppApplicationTests.storeAndRetrieveMovies.approved.txt ×

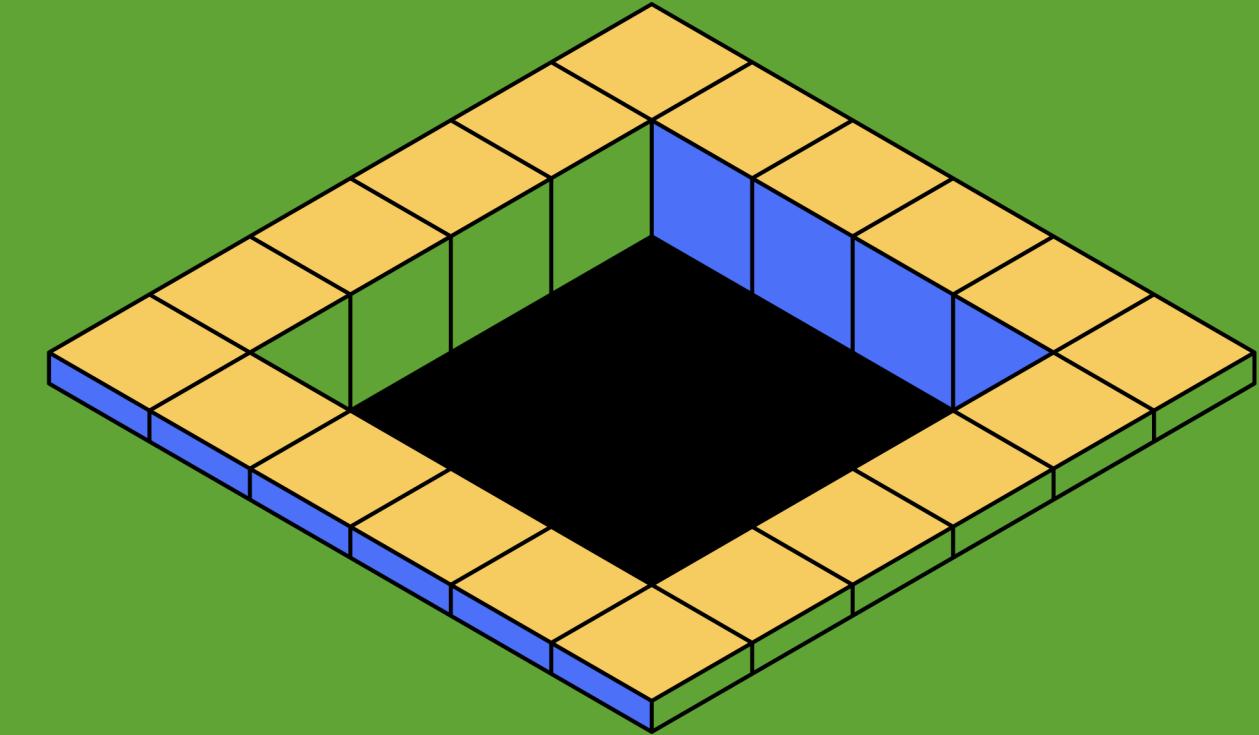
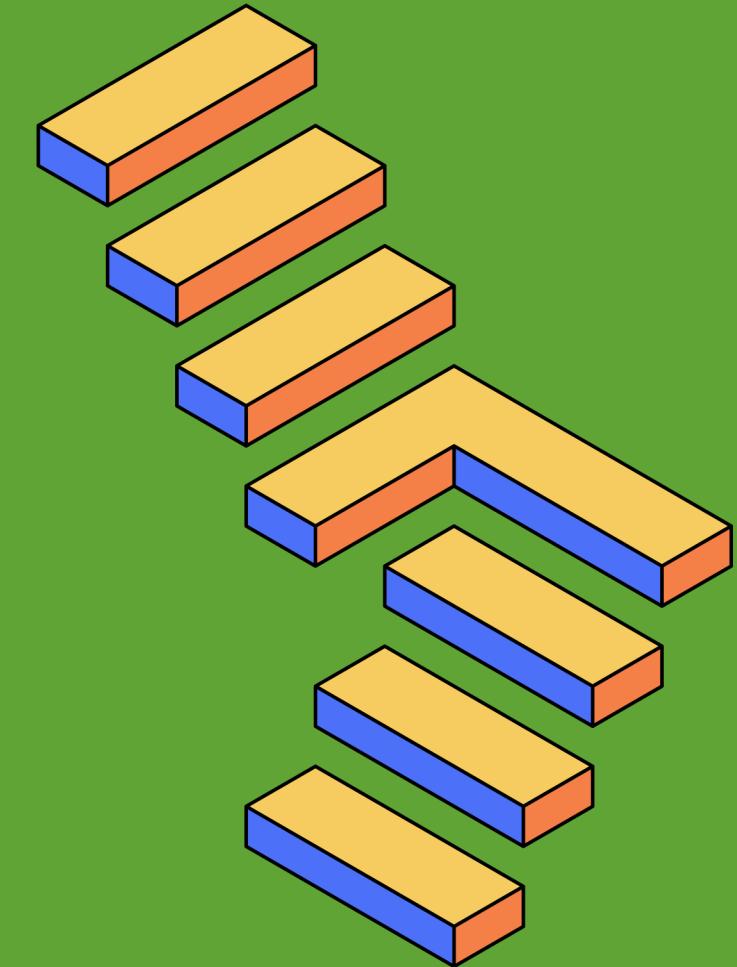
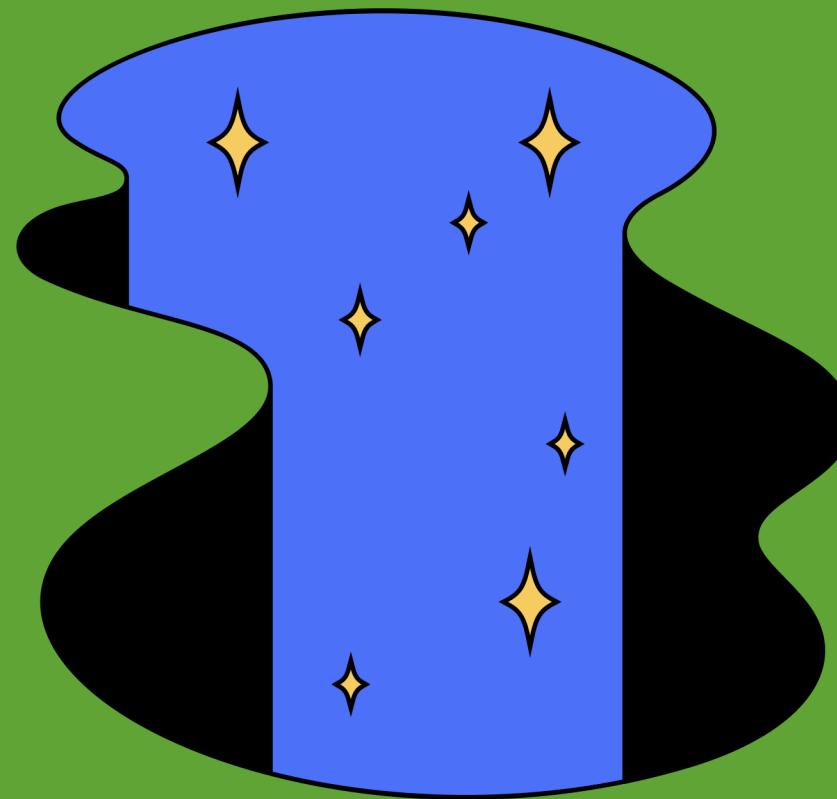
```
1 [ {
2   "title": "The Super Mario Bros. Movie",
3   "year": 2023
4 }, {
5   "title": "The Matrix",
6   "year": 1999
```



# Give it a try!



# Thank You!



**EMAIL**

[lars.eckart@hey.com](mailto:lars.eckart@hey.com)

**GitHub**

<https://github.com/LarsEckart>

**Web**

<https://larseckart.com/>