Acceptance and Integration Testing

Prof. Dr. Dirk Riehle

Friedrich-Alexander University Erlangen-Nürnberg

ADAP B03

Types of Tests [1] (Recap)

- **Components tests** (a.k.a. unit tests)
  - Focus on testing one component out of context
- **Acceptance tests** (a.k.a. functional tests)
  - Focus on testing one cross-cutting functionality
- **Integration tests** (a.k.a. system tests)
  - Focus on testing end-to-end system integrity

[1] This is a simplification for the purposes of this course.

Acceptance Tests

- Object under test is the system or a non-trivial subsystem
  - This is in contrast to unit testing, which isolates one component
- The tests focus on the system’s observable functionality
  - The PRD (product backlog) serves as the specification
- Test set-up has to cordon off rest of the system

Tell-a-Friend Acceptance Test

```java
public void testTellFriendMakeWebPart() {
   Map<String, String> args = new HashMap<String, String>();
   ...
   args.put(TellFriendFormHandler.EMAIL_SUBJECT, "Oh well...");
   handler.handlePost(session, args);

   part = handler.makeWebPart(session);
   assertEquals(part.getValue(TFFH.EMAIL_SUBJECT), "Oh well...");
}

public void testTellFriendPost() {
   EmailAddress from = EmailAddress.getFromString("i@w.org");
   EmailAddress to = EmailAddress.getFromString("fan@yahoo.com");
   String subject = "Coolest website ever!";
   ...
   Map<String, String> args = new HashMap<String, String>();
   args.put(TellFriendFormHandler.EMAIL_FROM, from.asString());
   args.put(TellFriendFormHandler.EMAIL_TO, to.asString());
   args.put(TellFriendFormHandler.EMAIL_SUBJECT, subject);
   args.put(TellFriendFormHandler.EMAIL_BODY, body);

   handler.handlePost(session, args);
}
```

Test Set-up Example (JUnit 3.8)

```
public class HandlerTestSetup extends TestSetup {
   public UserSession session;

   protected void setUp() throws Exception {
      super.setUp();
      session = createUserSession();
      ContextManager.setThreadLocalContext(session);

      Test test = getTest();
      if (test instanceof HandlerTest) {
         HandlerTest handlerTest = (HandlerTest) test;
         handlerTest.setUserSession(session);
      }
   }

   protected UserSession createUserSession() {
      Wahlzeit.configurePartHandlers();
      UserSession result = new UserSession("testContext");
      ...
      result.setConfiguration(LanguageConfigs.get(Language.ENGLISH));
      return result;
   }
}
```

How to Write Acceptance Tests

- Think from specification (through user interface)
- Sequentially test all relevant parameters
- Cover all functional edge cases

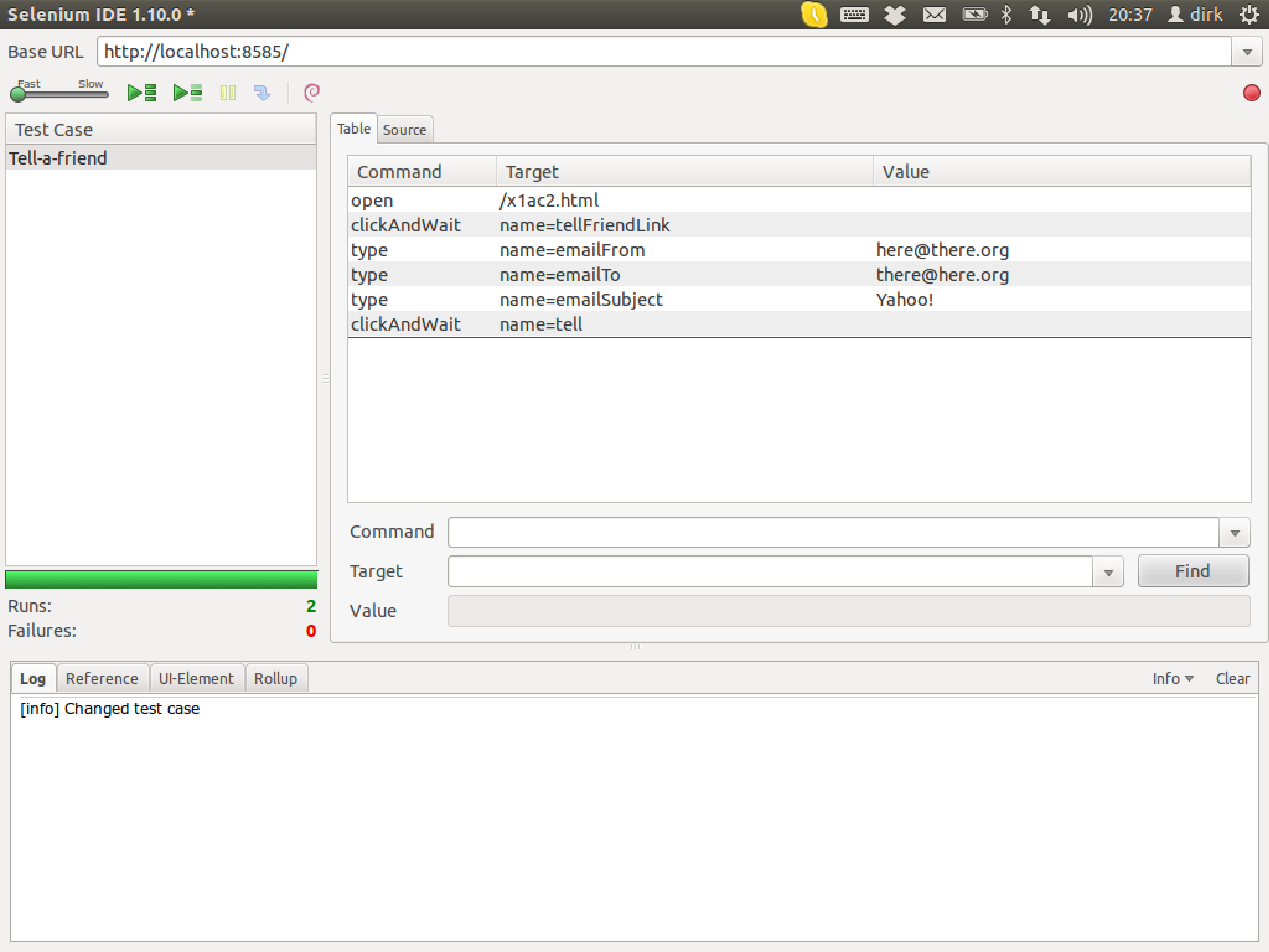
Tell-a-Friend Acceptance Test Example

```
public void testTellFriendMakeWebPart() {
   Map<String, String> args = new HashMap<String, String>();
   ...
   args.put(TellFriendFormHandler.EMAIL_SUBJECT, [illegible] well...");
   handler.handlePost(session, args);

   part = handler.makeWebPart(session);
   assertEquals(part.getValue([illegible].EMAIL_SUBJECT), "Oh well...");
}

public void testTellFriendPost() {
   EmailAddress from = EmailAddress.getFromString("[illegible]@w.org");
   EmailAddress to = EmailAddress.getFromString("[illegible]n@yahoo.com");
   String subject = "Coolest website ever";
   ...
   Map<String, String> args = new HashMap<String, String>();
   args.put(TellFriendFormHandler.EMAIL_FROM, from.asString());
   args.put(TellFriendFormHandler.EMAIL_TO, to.asString());
   args.put(TellFriendFormHandler.EMAIL_SUBJECT, subject);
   args.put(TellFriendFormHandler.EMAIL_BODY, body);

   handler.handlePost(session, args);
}
```

RATHER PAINFUL

Model-View-Separation and Testing

- Model-view-separation
    - Cleanly separates the domain model from it user interface(s)
    - Is a common simplification of the MVC pattern
    - Significantly simplifies functional testing of domain model
- Programmatic testing needs a clean model interface (API)
    - API = application programming interface
    - Wahlzeit provides a clean in-Java interface
    - Better would be a language independent API



Base URL http://localhost:8585/

Test Case
Tell-a-friend

Command	Target	Value
open	/x1ac2.html	
clickAndWait	name=tellFriendLink	
type	name=emailFrom	here@there.org
type	name=emailTo	there@here.org
type	name=emailSubject	Yahoo!
clickAndWait	name=tell	

Command
Target
Value

Runs: 2
Failures: 0

```
[info] Changed test case
```

Advanced Testing Concepts (Recap)

- Handling complex system set-ups
  - Mocking, stubbing, nulling
  - Dependency injection
- Testing specific system aspects
  - Concurrency
  - Legacy code
- Test structure and practicality
  - Extent of tests run, run-time

The Addison-Wesley Signature Series
xUnit Test Patterns
Refactoring Test Code
Gerard Meszaros
A Martin Fowler Signature Book
Foreword by Martin Fowler

Review / Summary of Session

- Acceptance and integration tests
- Ways of implementing these tests
- Challenges of complex testing

Thanks! Questions?

dirk.riehle@fau.de – **http://osr.cs.fau.de**

dirk@riehle.org – http://dirkriehle.com – @dirkriehle

Credits and License

- Original version
  - © 2012-2018 Dirk Riehle, some rights reserved
  - Licensed under Creative Commons Attribution 4.0 International License
- Contributions
  - ...

Acceptance and Integration Testing

Prof. Dr. Dirk Riehle

Friedrich-Alexander University Erlangen-Nürnberg

ADAP B03

It is Friedrich-Alexander University Erlangen-Nürnberg – FAU, in short.

Corporate identity wants us to say “Friedrich-Alexander University”.

Types of Tests [1] (Recap)

- **Components tests** (a.k.a. unit tests)
  - Focus on testing one component out of context
- **Acceptance tests** (a.k.a. functional tests)
  - Focus on testing one cross-cutting functionality
- **Integration tests** (a.k.a. system tests)
  - Focus on testing end-to-end system integrity

[1] This is a simplification for the purposes of this course.

Acceptance Tests

- Object under test is the system or a non-trivial subsystem
  - This is in contrast to unit testing, which isolates one component
- The tests focus on the system’s observable functionality
  - The PRD (product backlog) serves as the specification
- Test set-up has to cordon off rest of the system

Tell-a-Friend Acceptance Test

```
public void testTellFriendMakeWebPart() {
   Map<String, String> args = new HashMap<String, String>();
   ...
   args.put(TellFriendFormHandler.EMAIL_SUBJECT, "Oh well...");
   handler.handlePost(session, args);

   part = handler.makeWebPart(session);
   assertEquals(part.getValue(TFFH.EMAIL_SUBJECT), "Oh well...");
}

public void testTellFriendPost() {
   EmailAddress from = EmailAddress.getFromString("i@w.org");
   EmailAddress to = EmailAddress.getFromString("fan@yahoo.com");
   String subject = "Coolest website ever!";
   ...
   Map<String, String> args = new HashMap<String, String>();
   args.put(TellFriendFormHandler.EMAIL_FROM, from.asString());
   args.put(TellFriendFormHandler.EMAIL_TO, to.asString());
   args.put(TellFriendFormHandler.EMAIL_SUBJECT, subject);
   args.put(TellFriendFormHandler.EMAIL_BODY, body);

   handler.handlePost(session, args);
}
```

Test Set-up Example (JUnit 3.8)

```
public class HandlerTestSetup extends TestSetup {
   public UserSession session;

   protected void setUp() throws Exception {
      super.setUp();
      session = createUserSession();
      ContextManager.setThreadLocalContext(session);

      Test test = getTest();
      if (test instanceof HandlerTest) {
         HandlerTest handlerTest = (HandlerTest) test;
         handlerTest.setUserSession(session);
      }
   }

   protected UserSession createUserSession() {
      Wahlzeit.configurePartHandlers();
      UserSession result = new UserSession("testContext");
      ...
      result.setConfiguration(LanguageConfigs.get(Language.ENGLISH));
      return result;
   }
}
```

How to Write Acceptance Tests

- Think from specification (through user interface)
- Sequentially test all relevant parameters
- Cover all functional edge cases

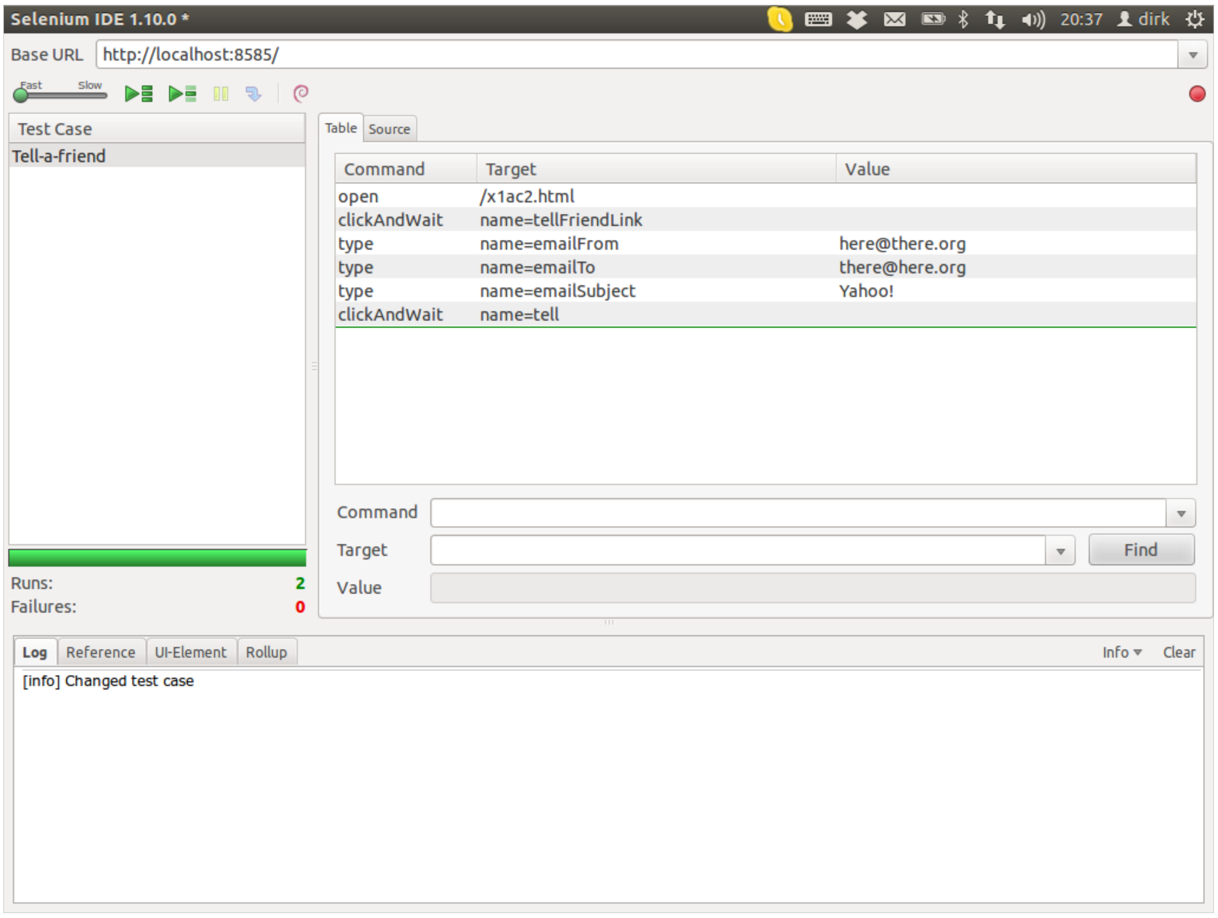
Tell-a-Friend Acceptance Test Example

```
public void testTellFriendMakeWebPart() {
   Map<String, String> args = new HashMap<String, String>();
   ...
   args.put(TellFriendFormHandler.EMAIL_SUBJECT, "Oh well...");
   handler.handlePost(session, args);

   part = handler.makeWebPart(session);
   assertEquals(part.getValue([illegible]n.EMAIL_SUBJECT), "Oh well...");
}

public void testTellFriendPost() {
   EmailAddress from = EmailAddress.getFromString("[illegible]@w.org");
   EmailAddress to = EmailAddress.getFromString("[illegible]an@yahoo.com");
   String subject = "Coolest website ever";
   ...
   Map<String, String> args = new HashMap<String, String>();
   args.put(TellFriendFormHandler.EMAIL_FROM, from.asString());
   args.put(TellFriendFormHandler.EMAIL_TO, to.asString());
   args.put(TellFriendFormHandler.EMAIL_SUBJECT, subject);
   args.put(TellFriendFormHandler.EMAIL_BODY, body);

   handler.handlePost(session, args);
}
```

RATHER PAINFUL

Model-View-Separation and Testing

- Model-view-separation
  - Cleanly separates the domain model from it user interface(s)
  - Is a common simplification of the MVC pattern
  - Significantly simplifies functional testing of domain model
- Programmatic testing needs a clean model interface (API)
  - API = application programming interface
  - Wahlzeit provides a clean in-Java interface
  - Better would be a language independent API



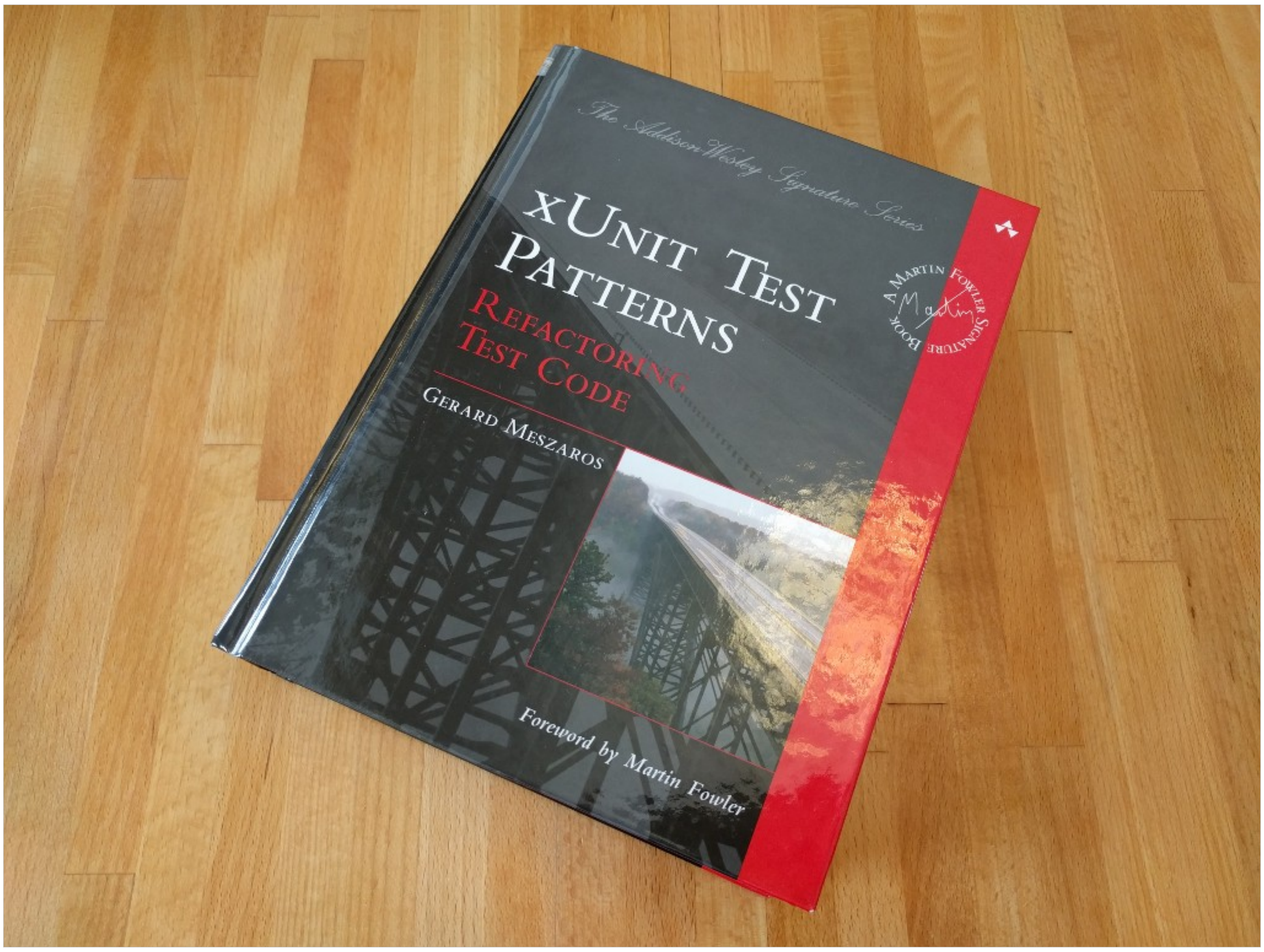
http://localhost:8585/

Command	Target	Value
open	/x1ac2.html	
clickAndWait	name=tellFriendLink	
type	name=emailFrom	here@there.org
type	name=emailTo	there@here.org
type	name=emailSubject	Yahoo!
clickAndWait	name=tell	

```
[info] Changed test case
```

Advanced Testing Concepts (Recap)

- Handling complex system set-ups
  - Mocking, stubbing, nulling
  - Dependency injection
- Testing specific system aspects
  - Concurrency
  - Legacy code
- Test structure and practicality
  - Extent of tests run, run-time


The Addison-Wesley Signature Series
xUnit Test Patterns
Refactoring Test Code
Gerard Meszaros
A Martin Fowler Signature Book
Foreword by Martin Fowler

Review / Summary of Session

- Acceptance and integration tests
- Ways of implementing these tests
- Challenges of complex testing

Thanks! Questions?

dirk.riehle@fau.de – http://osr.cs.fau.de

dirk@riehle.org – http://dirkriehle.com – @dirkriehle

Credits and License

- Original version
  - © 2012-2018 Dirk Riehle, some rights reserved
  - Licensed under Creative Commons Attribution 4.0 International License
- Contributions
  - ...