Jekyll and Hyde with Jubula

Alex Schladebeck
Markus Tiede

BREDEX GmbH

Agenda

- Introductions and motivation
- Dr Jekyll: the Jubula ITE
- Mr Hyde: the Jubula Client API

Introductions

Why do we test?

http://www.freeimages.com brainlocID 418215

Why do we automate tests?

http://www.freedigitalphotos.net Chaiwat 100300308

Automating tests with Jubula

- Code-free functional (black box) testing
  - Swing, SWT/RCP/GEF, HTML, iOS, JavaFX
- Tests *through* the GUI
  - No / minimal difference to manual test
  - All layers tested
  - Use cases, workflows: begins with requirements
  - Intelligent and well-structured tests
- Communication!
  - Whole team involvement
  - Continuous Integration = Continuous Information
- "Why can't we also write code…?"

CAP: the quintessence of automation

CAP (Test Step)

Component	Action	Parameter
(Which)	(What)	(How)

Username

Into the username field, enter "admin"

Demo ITE

Jubula ITE

Create test

Add data

Add components

Do mapping

Execute

Analyse

Jubula Client API

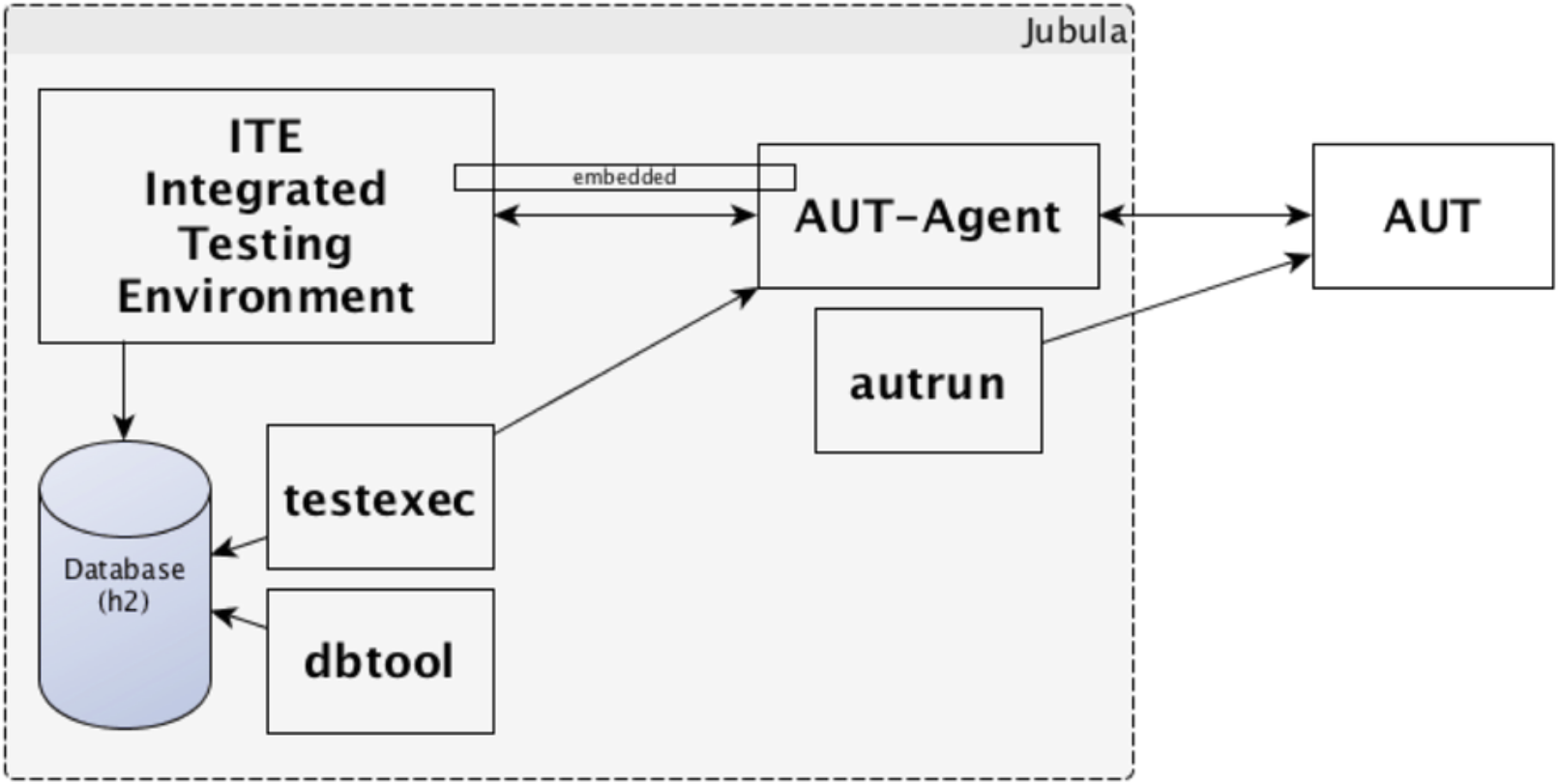
Java API

Jubula's Core

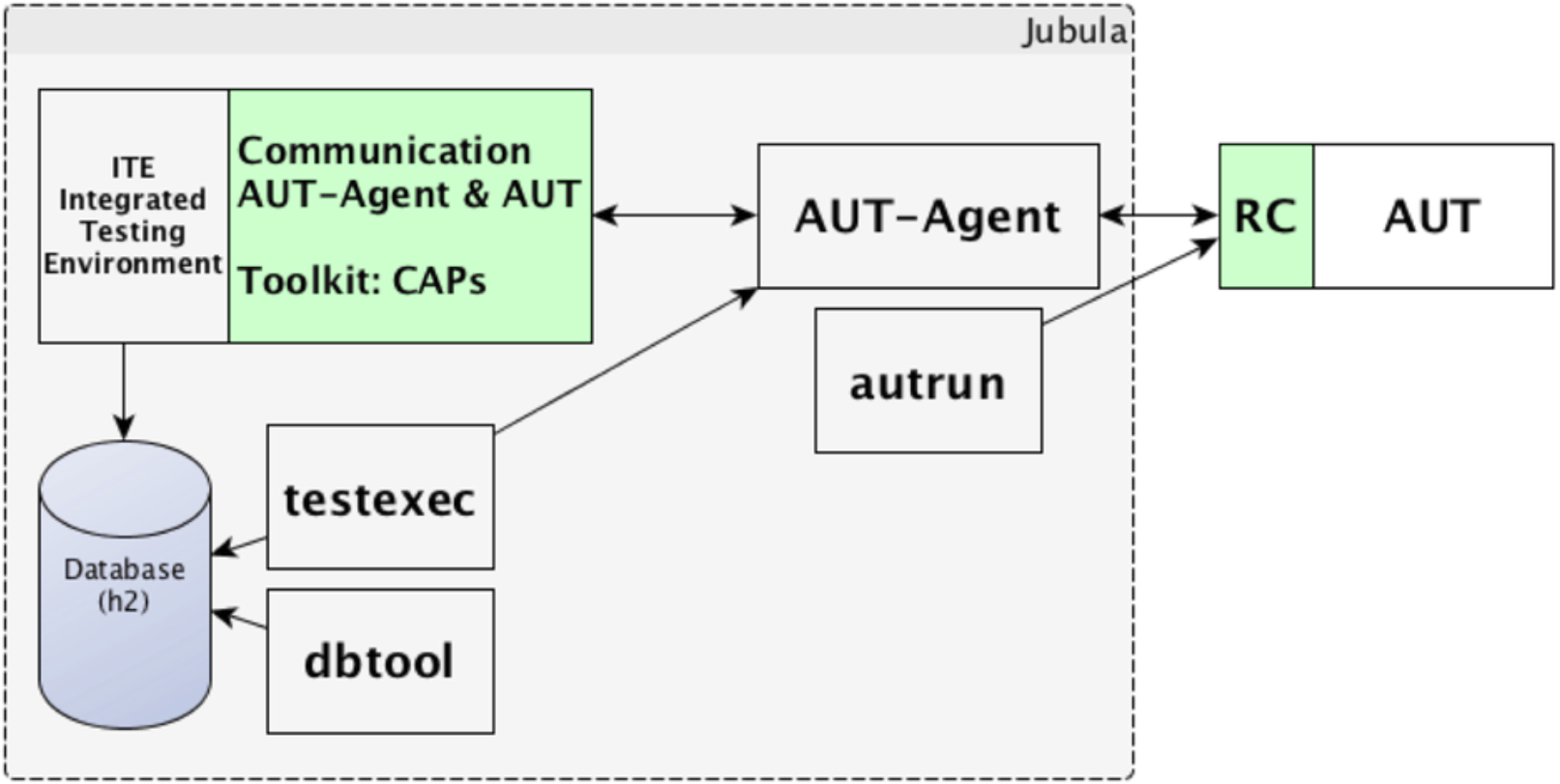
"Raw" low-level

AUT lifecycle
execute CAPs

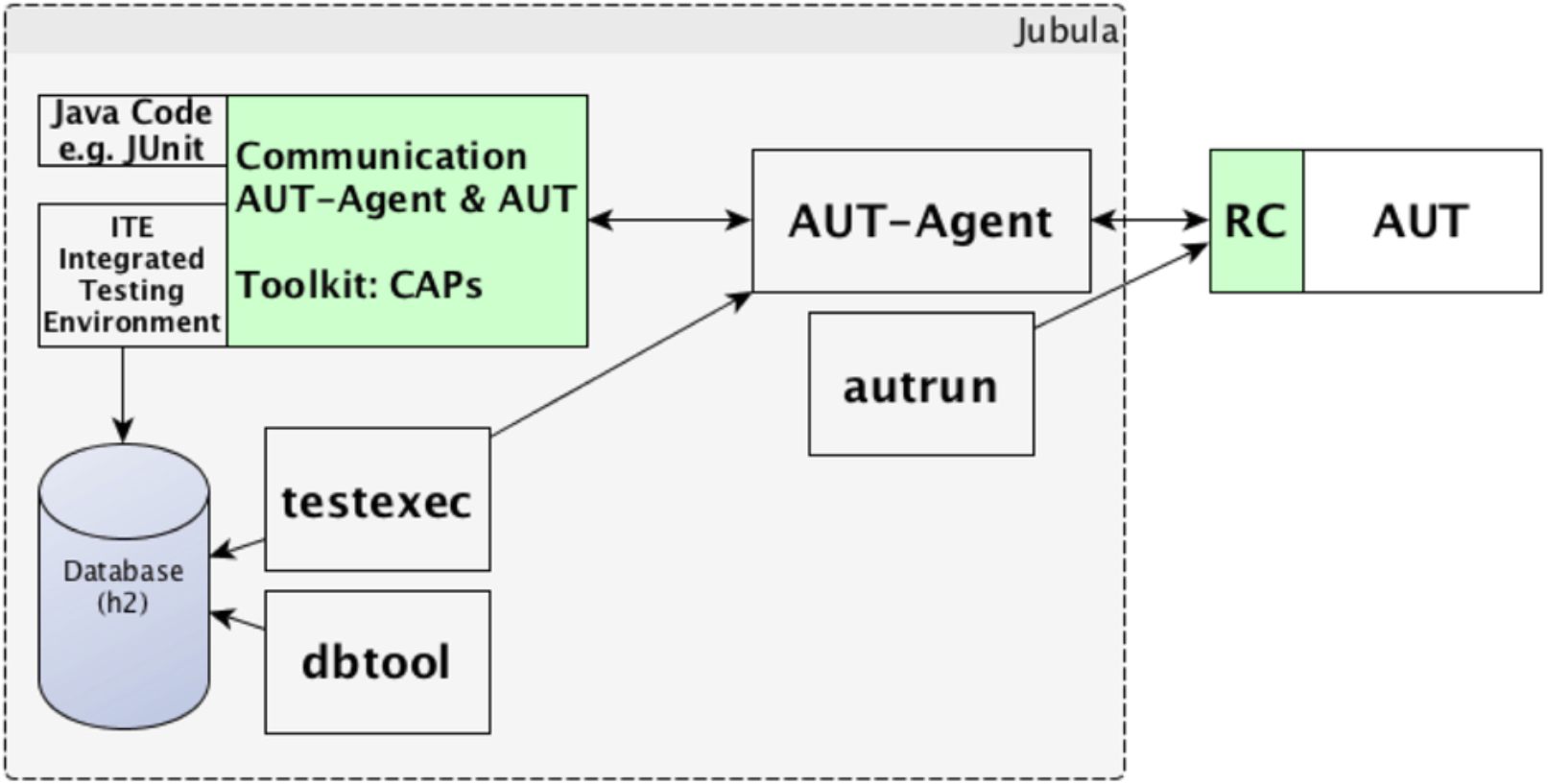
Let's spend some time with Mr Hyde



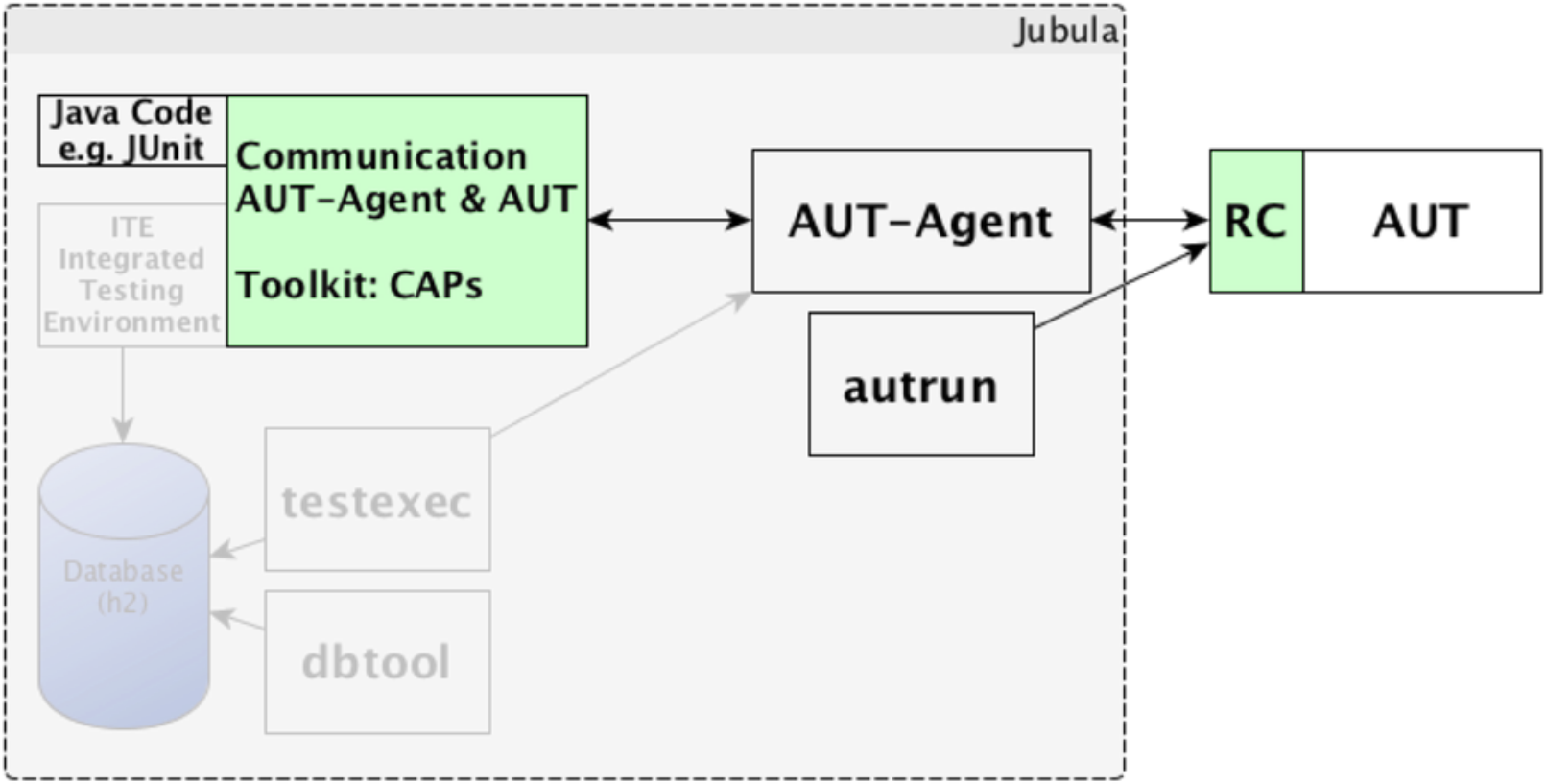
Let's spend some time with Mr Hyde



Let's spend some time with Mr Hyde



Let's spend some time with Mr Hyde



The AUT's lifecycle

```
/** prepare */
@Before
public void setUp() throws Exception {
    m_agent = MakeR.createAUTAgent(AGENT_HOST, AGENT_PORT);
    m_agent.connect();

    final String autID = "SimpleAdder_rcp";  //$NON-NLS-1$
    AUTConfiguration config = new RCPAUTConfiguration(
        "api.aut.conf.simple.adder.rcp",  //$NON-NLS-1$
        autID,
        "SimpleAdder.exe", //$NON-NLS-1$
        "..\\examples\\AUTs\\SimpleAdder\\rcp\\win32\\win32\\x86\\", //$NON-NLS-1$
        null,
        Locale.getDefault(),
        Locale.getDefault());

    AUTIdentifier id = m_agent.startAUT(config);
    if (id != null) {
        m_aut = m_agent.getAUT(id, SwtComponents
                .getToolkitInformation());
        m_aut.connect();
    } else {
        Assert.fail("AUT start has failed!"); //$NON-NLS-1$
    }
}
```

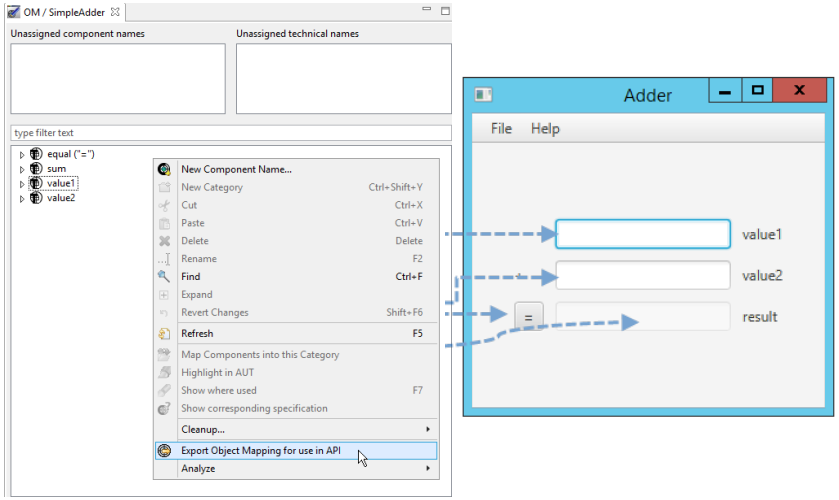
The AUT's lifecycle

```
/** prepare */
@Before
public void setUp() throws Exception {
    m_agent = MakeR.createAUTAgent(AGENT_HOST, AGENT_PORT);
    m_agent.connect();

    final String autID = "SimpleAdder_rcp";  //$NON-NLS-1$
    AUTConfiguration config = new RCPAUTConfiguration(
        "api.aut.conf.simple.adder.rcp",  //$NON-NLS-1$
        autID,
        "SimpleAdder.exe", //$NON-NLS-1$
        "..\\examples\\AUTs\\SimpleAdder\\rcp\\win32\\win32\\x86\\", //$NON-NLS-1$
        null,
        Locale.getDefault(),
        Locale.getDefault());

    AUTIdentifier id = m_agent.startAUT(config);
    if (id != null) {
        m_aut = m_agent.getAUT(id, SwtComponents
                .getToolkitInformation());
        m_aut.connect();
    } else {
        Assert.fail("AUT start has failed!"); //$NON-NLS-1$
    }
}
```

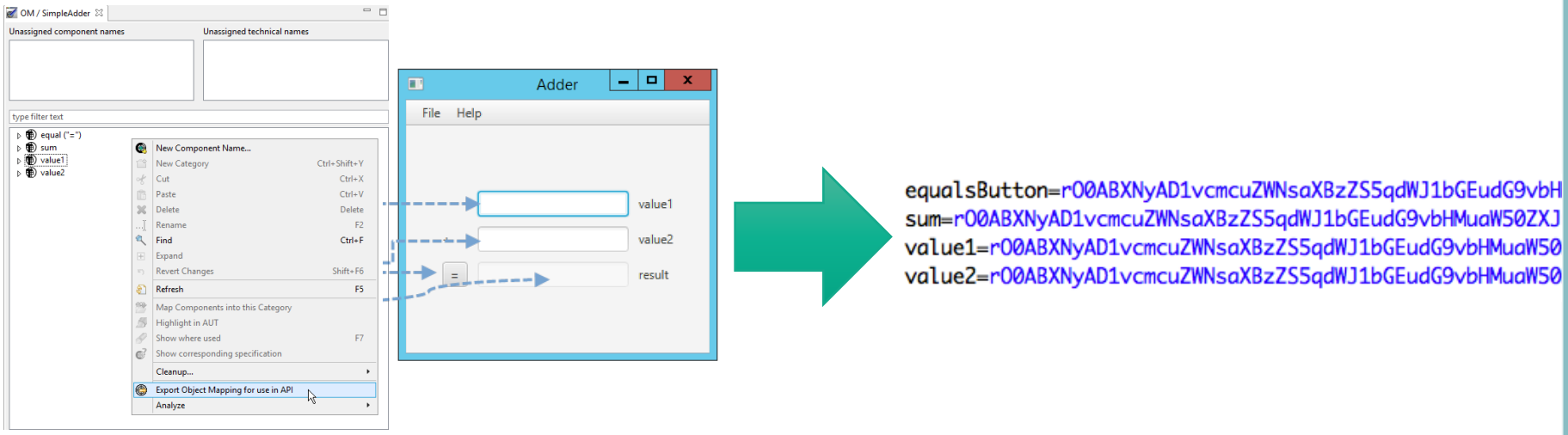
The AUT's lifecycle

```
/** prepare */
@Before
public void setUp() throws Exception {
    m_agent = MakeR.createAUTAgent(AGENT_HOST, AGENT_PORT);
    m_agent.connect();

    final String autID = "SimpleAdder_rcp";  //$NON-NLS-1$
    AUTConfiguration config = new RCPAUTConfiguration(
        "api.aut.conf.simple.adder.rcp",  //$NON-NLS-1$
        autID,
        "SimpleAdder.exe", //$NON-NLS-1$
        "..\\examples\\AUTs\\SimpleAdder\\rcp\\win32\\win32\\x86\\", //$NON-NLS-1$
        null,
        Locale.getDefault(),
        Locale.getDefault());

    AUTIdentifier id = m_agent.startAUT(config);
    if (id != null) {
        m_aut = m_agent.getAUT(id, SwtComponents
                .getToolkitInformation());
        m_aut.connect();
    } else {
        Assert.fail("AUT start has failed!"); //$NON-NLS-1$
    }
}
```

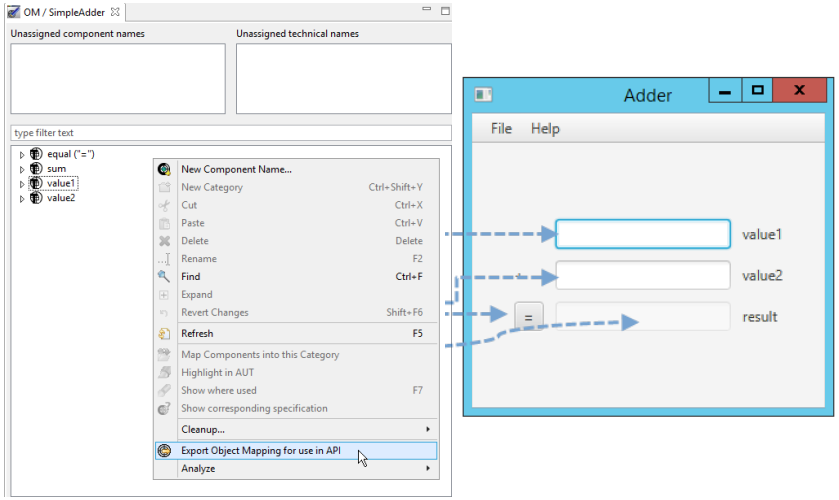
Accessing a UI component

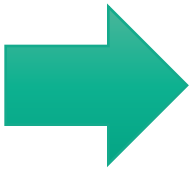


Accessing a UI component



Accessing a UI component



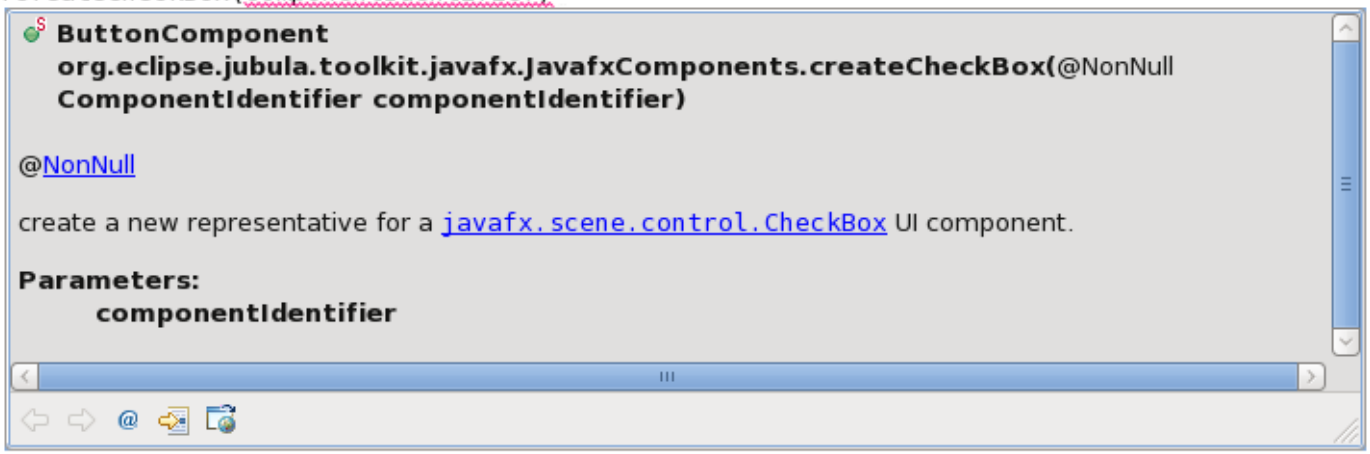


```
equalsButton=rO0ABXNyAD1vcmcuZWNsaXBzZS5qdWJ1bGEudG9vbH
sum=rO0ABXNyAD1vcmcuZWNsaXBzZS5qdWJ1bGEudG9vbHMuaW50ZXJ
value1=rO0ABXNyAD1vcmcuZWNsaXBzZS5qdWJ1bGEudG9vbHMuaW50
value2=rO0ABXNyAD1vcmcuZWNsaXBzZS5qdWJ1bGEudG9vbHMuaW50
```

```
URL input = SimpleAdder.class.getClassLoader().getResource(
        "objectMapping_SimpleAdderRCP.properties"); //$NON-NLS-1$

ObjectMapping om = MakeR.createObjectMapping(input.openStream());

ComponentIdentifier<Text> val1Id = om.get("value1"); //$NON-NLS-1$
ComponentIdentifier<Text> val2Id = om.get("value2"); //$NON-NLS-1$
ComponentIdentifier<Button> buttonId = om.get("equalsButton"); //$NON-NLS-1$
ComponentIdentifier<Label> sumId = om.get("sum"); //$NON-NLS-1$
```

Accessing a UI component

```
JavafxComponents.createCheckBox(componentIdentifier)
```



- UI component proxy instances
  - factories for executable CAPs
  - searched when CAPs getting executed

Available for all toolkits

abstract

Textual components

Graphical components

concrete

Button components

Combo components

Table components

Tree components

List components

...

SWT

Swing

JavaFX

mobile

HTML

TreeTable

RCP & GEF

Canvas

TableView

TreeView

ButtonBase

ComboBoxBase

Stage

...

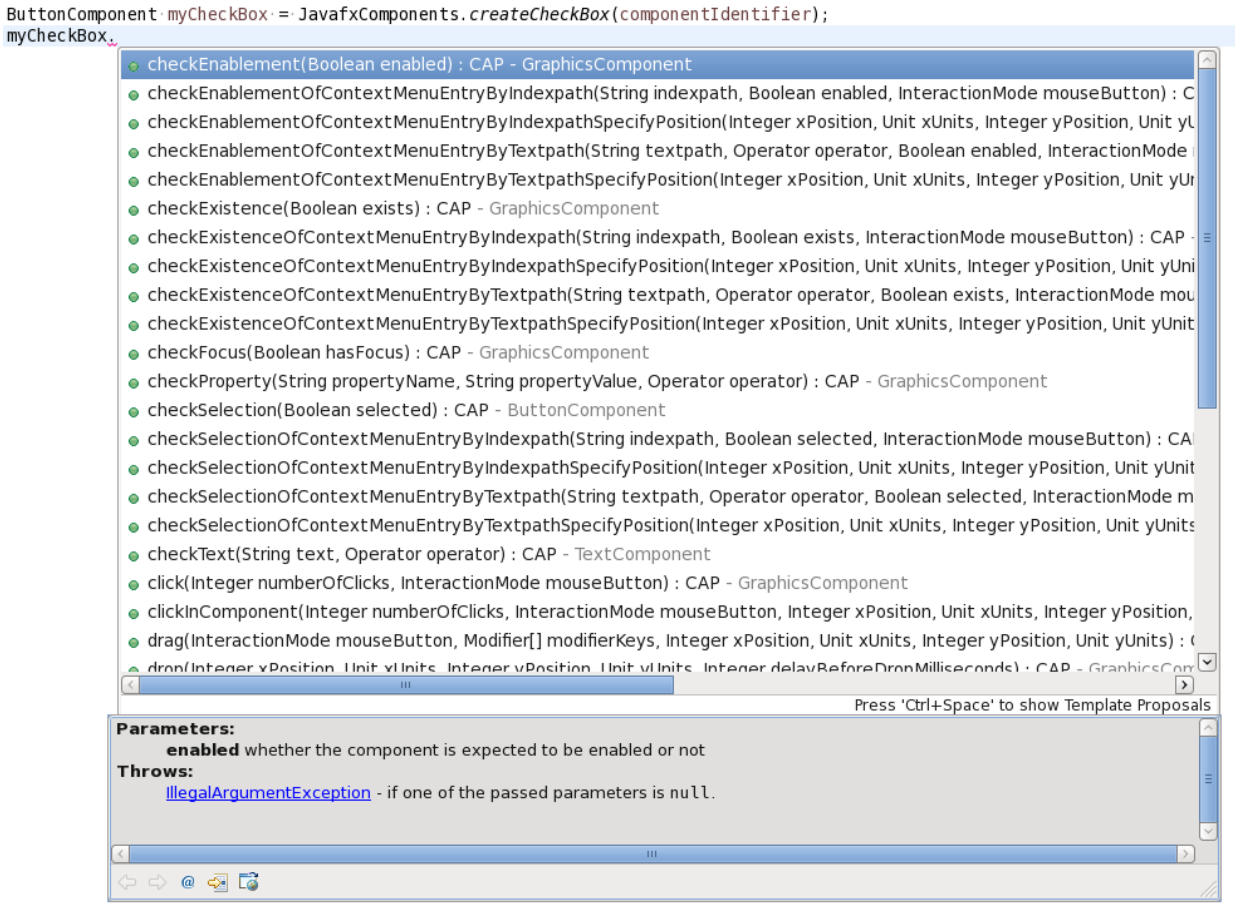
iOS

UIPicker

Gestures

Hyperlink

200+ CAPs / Test Steps



Executing CAPs

```
/** the actual test method */
@Test(expected = CheckFailedException.class)
public void testTestFirstSimpleAdderSteps() throws Exception {

    final int firstValue = 17;
    List<Result<String>> results = new ArrayList<Result<String>>();
    try {
        for (int i = 1; i < 5; i++) {
            results.add(m_aut.execute(
                    value1.replaceText(
                        String.valueOf(firstValue)),
                        "Entering first value")); //$NON-NLS-1$
            results.add(m_aut.execute(
                    value2.replaceText(
                        String.valueOf(i)),
                        "Entering second value")); //$NON-NLS-1$
            results.add(m_aut.execute(
                    button.click(
                        1, InteractionMode.primary),
                        "Invoking addition")); //$NON-NLS-1$
            results.add(m_aut.execute(
                    result.checkText(
                        String.valueOf(firstValue + i),
                        Operator.equals),
                        "Checking computed result")); //$NON-NLS-1$
        }
    } finally {
        Assert.assertTrue(results.size() == 15);
    }
}
```

Executing CAPs

```
/** the actual test method */
@Test(expected = CheckFailedException.class)
public void testTestFirstSimpleAdderSteps() throws Exception {

    final int firstValue = 17;
    List<Result<String>> results = new ArrayList<Result<String>>();
    try {
        for (int i = 1; i < 5; i++) {
            results.add(m_aut.execute(
                    value1.replaceText(
                        String.valueOf(firstValue)),
                        "Entering first value")); //$NON-NLS-1$
            results.add(m_aut.execute(
                    value2.replaceText(
                        String.valueOf(i)),
                        "Entering second value")); //$NON-NLS-1$
            results.add(m_aut.execute(
                    button.click(
                        1, InteractionMode.primary),
                        "Invoking addition")); //$NON-NLS-1$
            results.add(m_aut.execute(
                    result.checkText(
                        String.valueOf(firstValue + i),
                        Operator.equals),
                        "Checking computed result")); //$NON-NLS-1$
        }
    } finally {
        Assert.assertTrue(results.size() == 15);
    }
}
```

Executing CAPs

```
/** the actual test method */
@Test(expected = CheckFailedException.class)
public void testTestFirstSimpleAdderSteps() throws Exception {

    final int firstValue = 17;
    List<Result<String>> results = new ArrayList<Result<String>>();
    try {
        for (int i = 1; i < 5; i++) {
            results.add(m_aut.execute(
                    value1.replaceText(
                        String.valueOf(firstValue)),
                        "Entering first value")); //$NON-NLS-1$
            results.add(m_aut.execute(
                    value2.replaceText(
                        String.valueOf(i)),
                        "Entering second value")); //$NON-NLS-1$
            results.add(m_aut.execute(
                    button.click(
                        1, InteractionMode.primary),
                        "Invoking addition")); //$NON-NLS-1$
            results.add(m_aut.execute(
                    result.checkText(
                        String.valueOf(firstValue + i),
                        Operator.equals),
                        "Checking computed result")); //$NON-NLS-1$
        }
    } finally {
        Assert.assertTrue(results.size() == 15);
    }
}
```

Demo API

Some facts

- Java 6+
- Exceptions for execution events
- Consumable in different flavors
  - plain JARs
  - Maven artifacts from m2-repo
  - OSGi bundles
- Small footprint ~2MB
- Available since Mars
  - Standalone version: http://testing.bredex.de

What's happened since Mars?

- **New RC features:**
  - New actions for:
    - GEF
    - JavaFX
    - HTML
    - SWT/RCP
- **New ITE features:**
  - Quicker test result report opening
  - Updates for JavaFX
    - Autrun
    - Code coverage
    - New mapping options
- **New client API features:**
  - Custom UI controls
  - GEF via the API
  - Taking screenshots
  - Partial export of object mapping from the ITE
  - more example scenarios

Dual personalities

- Jubula ITE
  - Code-free
    - + test design
  - Communication
  - Offers shiny extras
    - ALM
    - Test result reports
    - BIRT
- Jubula Client API
  - Very code-y
    - + coding practices ;)
  - Communication
  - Offers shiny features
    - Complex test flows
    - Use the world of Java
    - Integration into SCM

Choose your poison!