

Типы автоматического тестирования в IntelliJ IDEA

Юрий Артамонов

Автор ты кто

Юрий Артамонов @jreznот

1. Разрабатывал фреймворки и библиотеки для Java > 10 лет
2. Придумываю новые возможности IDE для ваших любимых фреймворков в IntelliJ IDEA
3. Автор Selenium UI Testing плагина и мейнтейнер Gauge плагина для IntelliJ IDEA

Joker<?>

#jokerconf



План действий

1. Как устроены JetBrains IDE
2. Что тестировать в IDE
3. Unit & Functional тесты
4. Property-based тесты
5. UI тесты
6. Как работают команды IDE в JetBrains

Зачем вам слушать этот доклад?

1. Разработать плагин для JetBrains IDE
2. Посмотреть, а как тестируют другие
3. Узнать что-то новое про устройство IDE

The screenshot shows the IntelliJ IDEA IDE interface. On the left is the Project tool window showing a file tree with folders like 'extract', 'findUsages', 'folding', etc., and files like 'GaugeUtil', 'HookUtil', 'SocketUtils', 'StepUtil'. The main editor displays the code for 'GaugeSettingsService.java'. The code includes package, import, and class definitions. A yellow highlight is on the '@Storage' annotation. A green oval highlights the class definition and its methods. A yellow diamond highlights the 'GaugeSettingsService' class name. On the right is the Gradle tool window showing a task tree with 'Tasks' expanded to show 'build', 'clean', 'dependencies', etc.

```
1  /.../
2
3  package com.thoughtworks.gauge.settings;
4
5  import ...
6
7  @State(
8      name = "GaugeConfiguration",
9      storages = {
10         @Storage(value = "GaugeConfig.xml", roamingType = DISABLED),
11     }
12 )
13
14 public final class GaugeSettingsService implements PersistentStateComponent<GaugeSettingsModel> {
15     private GaugeSettingsModel state = new GaugeSettingsModel();
16
17     @Nullable
18     @Override
19     public GaugeSettingsModel getState() { return state; }
20
21     @Override
22     public void loadState(@NotNull GaugeSettingsModel state) { this.state = state; }
23
24     public static GaugeSettingsModel getSettings() {
25         GaugeSettingsService service = getService();
26         return service.getState();
27     }
28
29     public static GaugeSettingsService getService() { return ApplicationManager.getApplication().getService(GaugeSettingsService); }
30 }
```

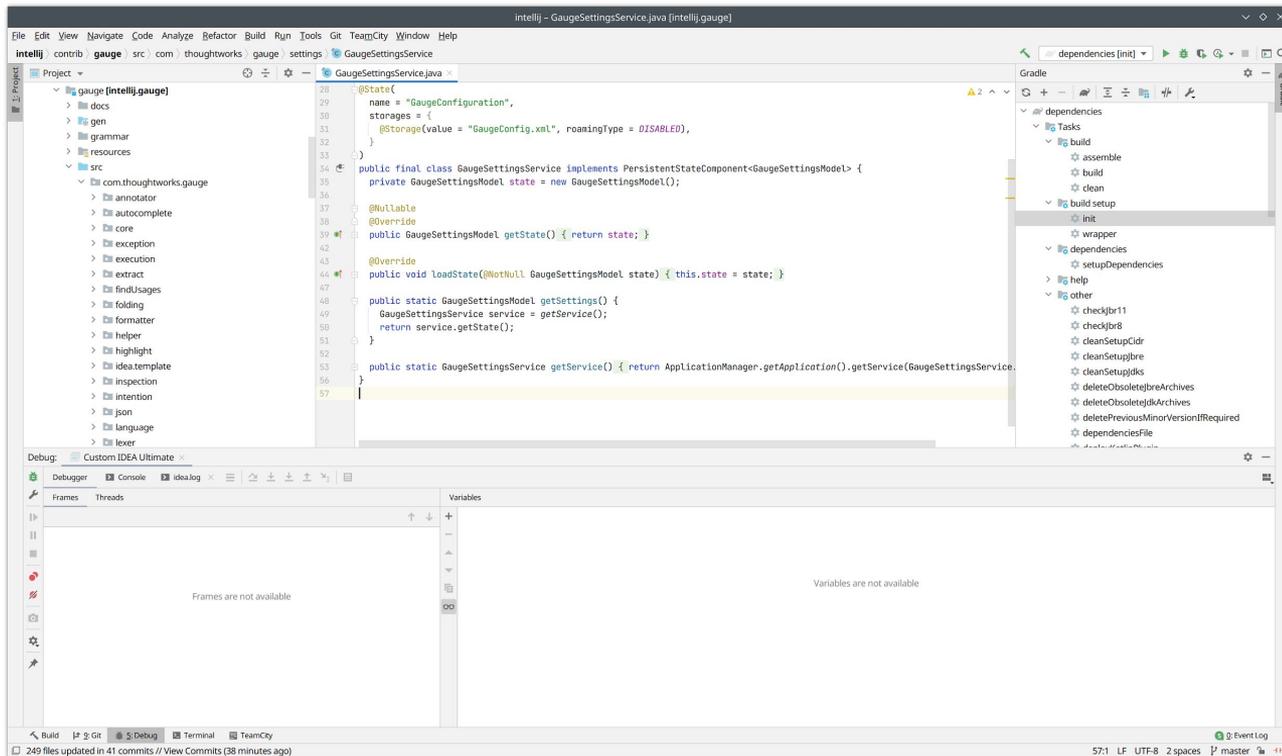
The Run tool window shows the output of a Gradle build. The build is successful and took 1 second and 81 milliseconds. The output includes the following tasks and their status:

```
dependencies [setupBundledMaven]: successful At 08.10.2020, 11:29    1 sec, 81 ms    11:29:11: Executing task 'setupBundledMaven'...
> Task :deletePreviousMinorVersionIfRequired SKIPPED
> Task :setupCommonLibs UP-TO-DATE
> Task :unpackMavenDistribution SKIPPED
> Task :setupBundledMaven SKIPPED

BUILD SUCCESSFUL in 239ms
1 actionable task: 1 up-to-date
11:29:12: Task execution finished 'setupBundledMaven'.
```

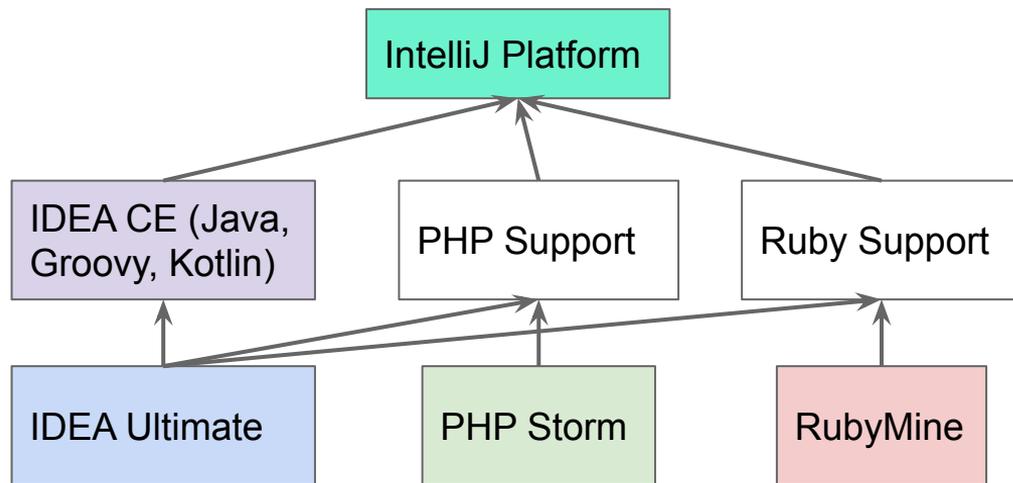
JetBrains IDE снаружи

- Menus
- Project View
- **Editor**
- Tool windows
- Settings
- Debugger
- VCS
- Status bar



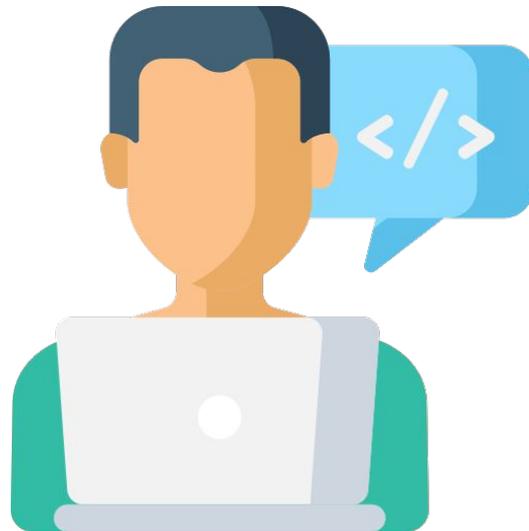
JetBrains IDE изнутри

- Компоненты и сервисы
- Виртуальная файловая система (VFS)
- Поддержка языков (PSI)
- Редактор кода
- Инспекции
- Индексы
- Фоновые процессы
- UI библиотека
- Точки расширения



Типовые тестовые сценарии

1. Создание проекта
2. Редактирование кода
3. Рефакторинг
4. Взаимодействие с VCS
5. Сборка
6. Запуск
7. Отладка



Модульные тесты и специфика IDE

1. Можно иногда
2. В IDE всё зависит от всего.

Пример: Selenium плагин
зависит от Java, CSS, XPath,
Docker, Maven и Gradle

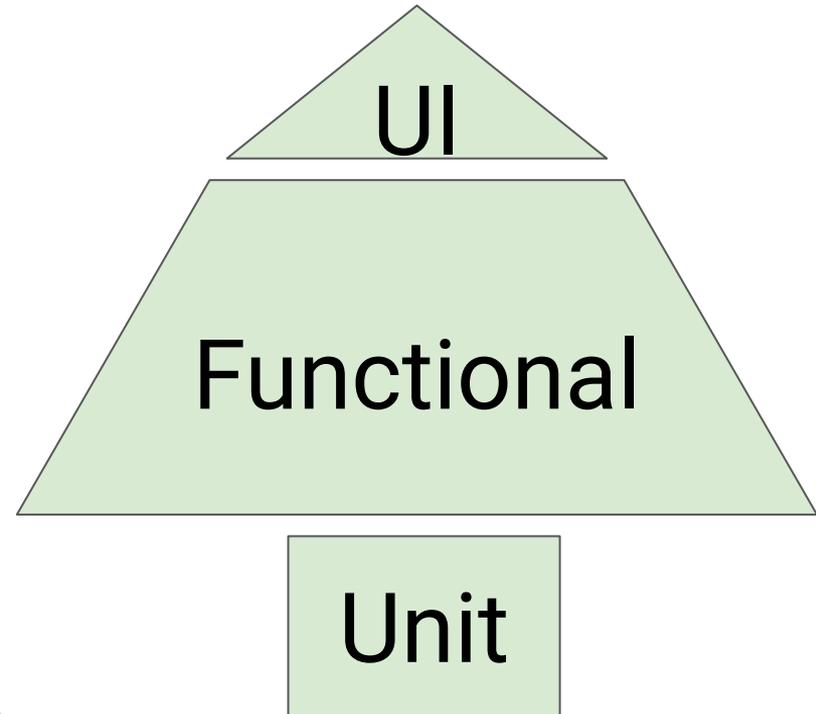
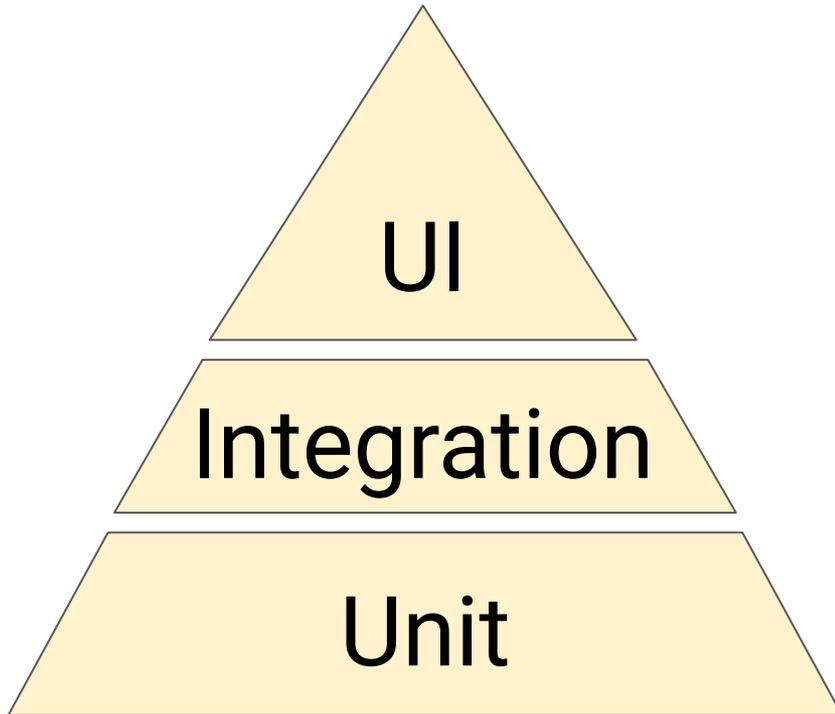


Функциональные тесты IDE

Какие тесты мы пишем чаще всего:

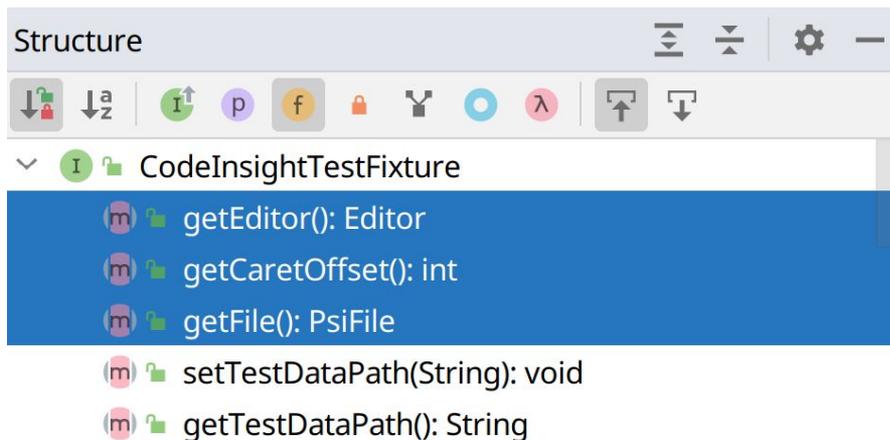
- Тесты запускаются в памяти без UI
- Тесты используют реальные реализации большинства подсистем IDE, за исключением UI компонентов
- Тесты проверяют один из сценариев использования целиком, а не отдельные функции и свойства реализации.

Пирамида или Ёлка



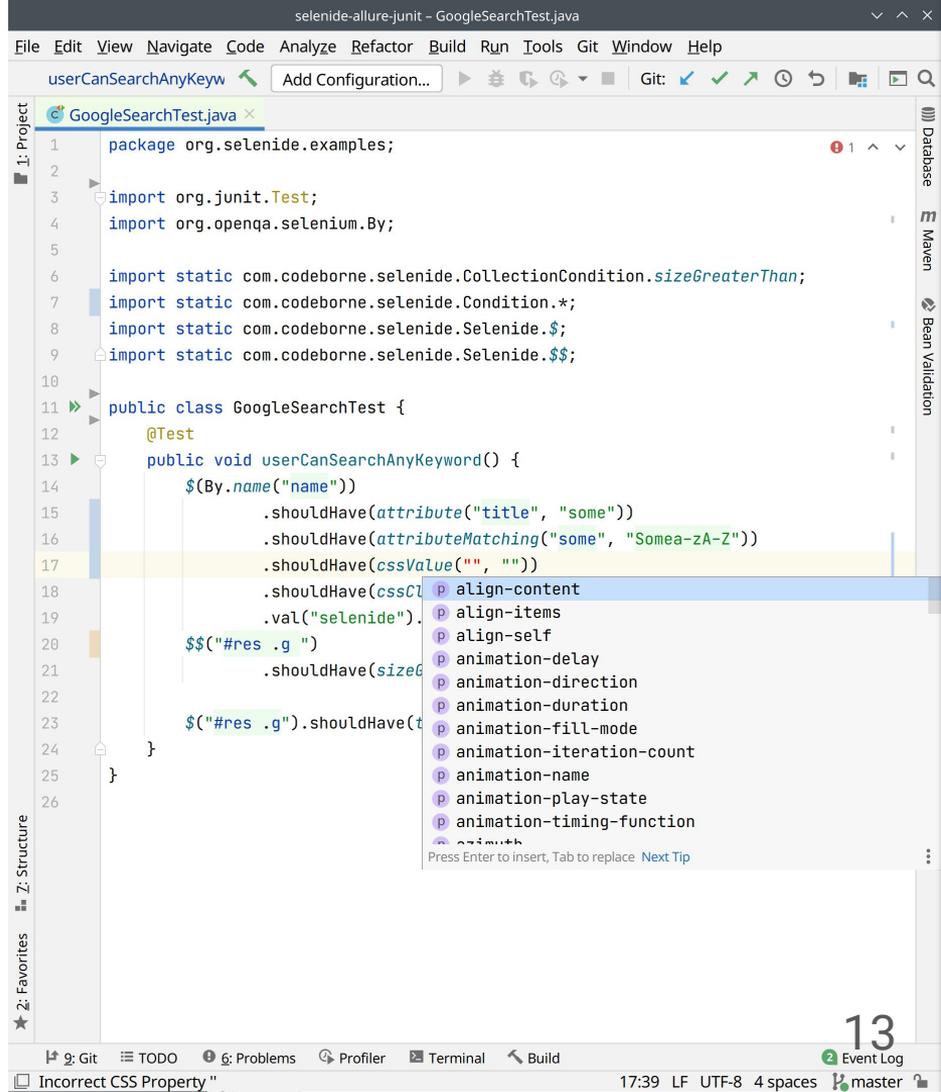
Fixture - модель IDE в памяти

- Тесты не тестируют UI, а работают с моделью IDE, как с интерфейсом.
- Большинство тестов принимают на вход файл/файлы, выполняют сценарий и сравнивают результат с эталоном.



На что писать функциональные тесты

1. Lexers / Parsers
2. Highlighting
3. Auto completion
4. Inspections
5. Intentions
6. References
7. Navigation / Find usages



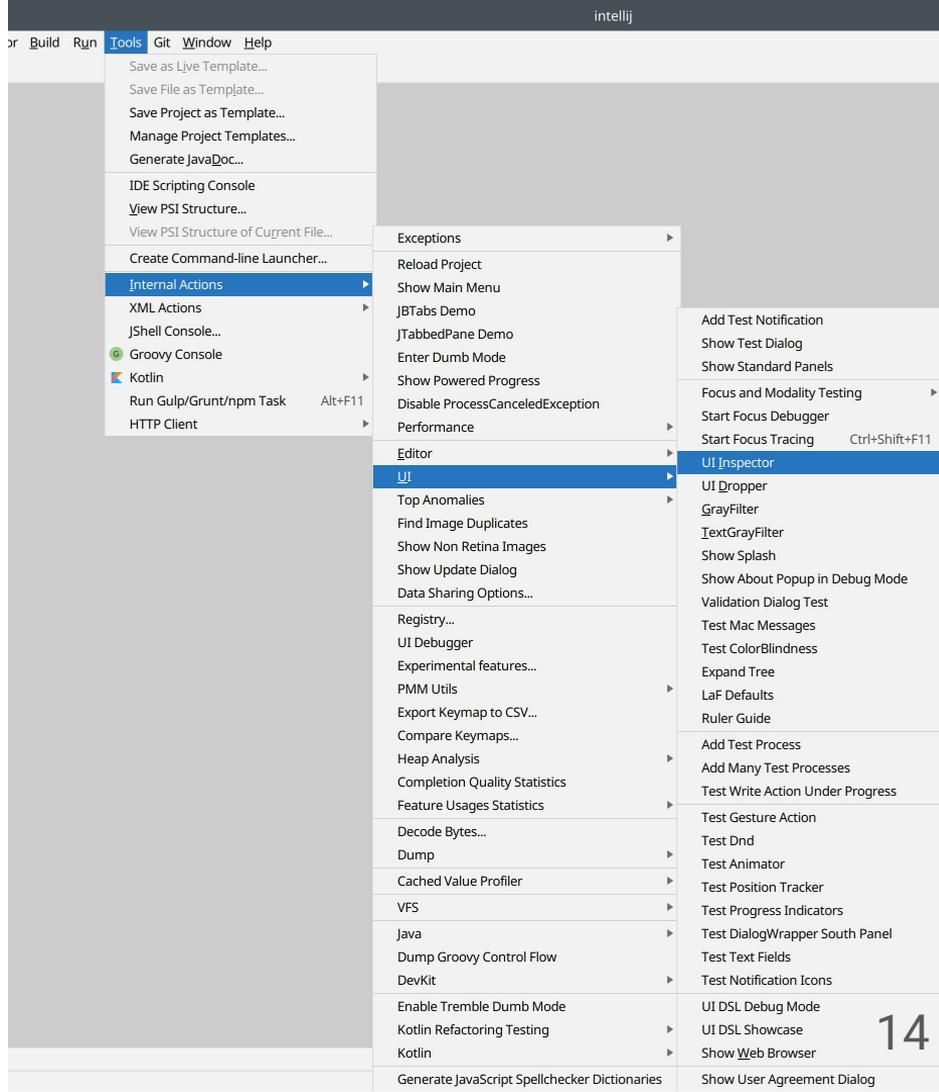
Internal Mode

Специальный режим работы IDE, в котором доступны опции для разработчиков.

- View PSI Structure of Current File
- Internal Actions
- UI Inspector
- Registry (только тссс...)

Включить в VM Options своей IDE:

`-Didea.is.internal=true`



PSI Viewer

Tools - View PSI Structure of Current File

Позволяет изучить структуру PSI
дерева в файле:

- Дерево PSI элементов
- Ссылки между элементами
- Встроенные фрагменты кода

The screenshot displays the PSI Viewer interface for a Java file. At the top, there are settings for 'Show PSI structure for:' (set to 'JAVA file'), 'Show PsiWhiteSpace' (checked), 'Show tree nodes' (checked), and 'Dialect:' (set to 'Java').

The main area shows the source code of a test method:

```
12 @Test
13 public void userCanSearchAnyKeyword() {
14     $(By.name("name"))
15         .shouldHave(attribute("title", "some"))
16         .shouldHave(attributeMatching("some", "Somea-zA-Z"))
17         .shouldHave(cssValue("font-size", ""))
18         .shouldHave(cssClass("no"))
19         .val("selenide").pressEnter();
20     $("#res .g ")
21         .shouldHave(sizeGreaterThan(5));
22
23     $("#res .g").shouldHave(text("selenide.org"));
24 }
25 }
26
```

Below the code is the 'PSI Tree' view, which shows a hierarchical structure of PSI elements. The tree is expanded to show the following structure:

- PsiMethodCallExpression:\$(By.name("name")) .shouldHave(e
- PsiReferenceExpression:\$(By.name("name")) .shouldHave
- PsiMethodCallExpression:\$(By.name("name")) .should
- PsiReferenceExpression:\$(By.name("name")) .shoul
- PsiMethodCallExpression:\$(By.name("name")) .s
- PsiReferenceExpression:\$(By.name("name"))
- PsiMethodCallExpression:\$(By.name("name"))
- PsiReferenceExpression:\$(By.name("name"))
- PsiExpressionList
 - PsijavaToken:LPARENTH
 - PsiMethodCallExpression:cssValue("font-siz
 - PsiReferenceExpression:cssValue
 - PsiExpressionList
 - PsijavaToken:LPARENTH
 - PsiLiteralExpression:"font-size"
 - PsijavaToken:STRING_LITERAL
 - PsijavaToken:COMMA

On the right side, there are tabs for 'References', 'Block Structure', and 'Stub Structure'. The 'References' tab is active, showing a reference to 'com.intellij.selenium.shared.dom.CssPropertyReference'.

At the bottom right, there is a '15' icon and a 'Close' button. At the bottom center, there is a 'Copy PSI' button and a 'Build PSI Tree' button.

Тестируем парсер

Базовый класс: `com.intellij.testFramework.ParsingTestCase`

1. Создаём тестовый метод:

```
public void testSpecWithDataTable() { doTest(true); }
```

2. Добавляем исходный файл: `SpecWithDataTable.spec`

3. И файл с ожидаемой структурой кода: `SpecWithDataTable.txt`

Демо

Light and Heavy Tests

Мы разделяем интеграционные тесты на:

- **Heavy Tests** - создают новый проект на каждый тест
- **Light Tests** - переиспользуют проекты между запусками, когда возможно

В основном, мы рекомендуем писать **Light Tests**, чтобы сократить время исполнения.

Каким Моск-фреймворком пользуются в IntelliJ IDEA ?

- JMockit
- Mockito
- EasyMock
- Custom



Никаким

Моки для тестирования плагинов

1. Моки сервисов IDE
2. Mock JDK
3. Классы из библиотек
4. Реальные зависимости

Демо



Тестирование зависимых плагинов

1. **Пример:** Spring плагин + Kotlin
Два комплекта тестов с разными плагинами в ClassPath IDE.
2. **Пример:** Java тесты в IntelliJ CE и IntelliJ Ultimate

Property-based тесты

Алгоритм:

1. Подготовить / сгенерировать сценарий из случайных действий
2. На каждом шаге сценария выполнить проверку свойства
3. При падении теста подготовить сериализованное представление сценария для воспроизведения



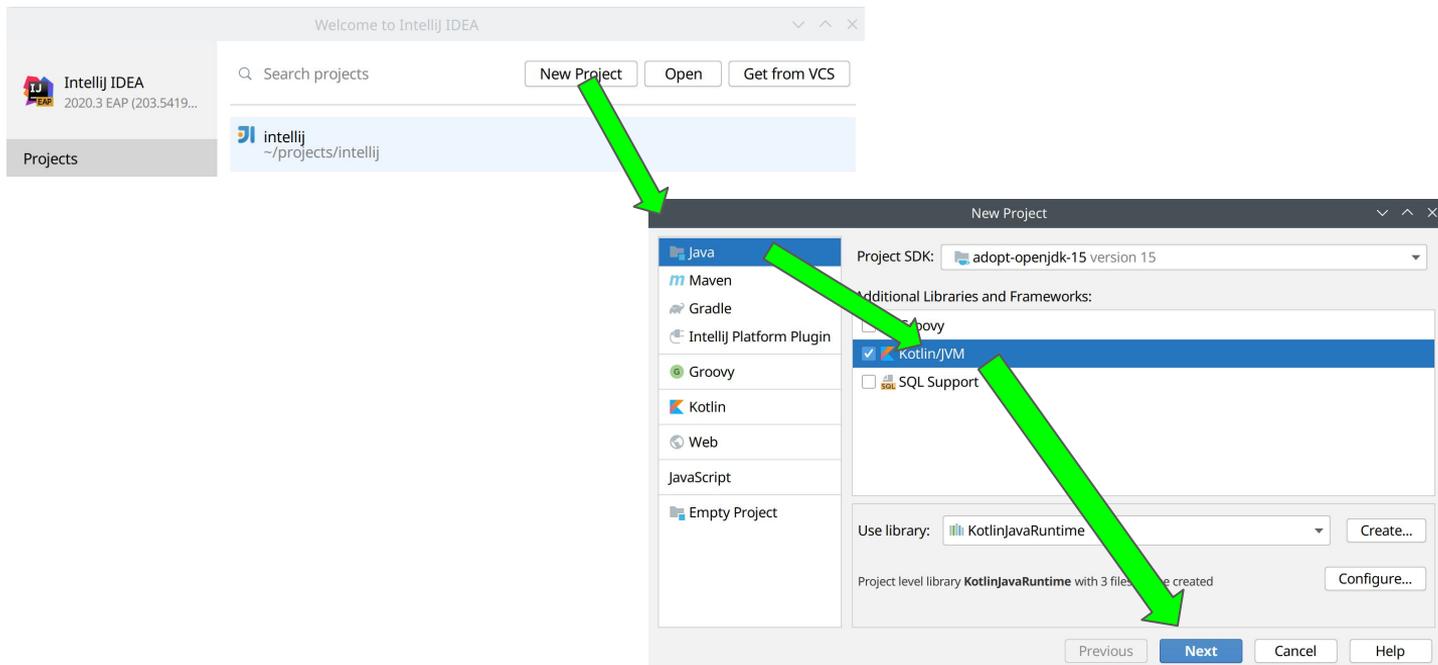
[JetBrains/jetCheck](https://github.com/JetBrains/jetCheck)

Примеры:

1. Вставить комментарии в случайные строки
2. Поменять тип всех методов на Object

E2E тестирование IDE

А как протестировать пользовательский сценарий целиком?



Selenium-like подход для тестирования UI

1. Чем плохи готовые решения?

По сравнению с web, в desktop почти нет инструментов для автоматизации UI.

2. Классический вариант тестирования Java UI:

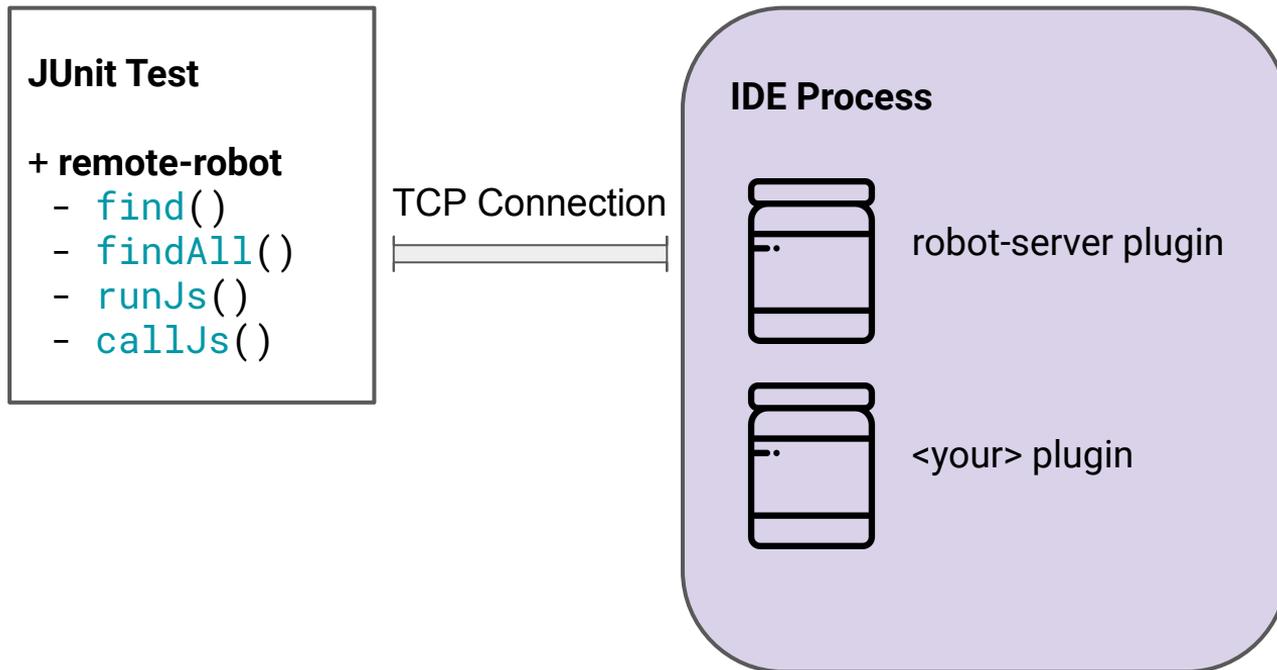
`java.awt.Robot` в runtime приложения / FEST-Swing



[JetBrains/intellij-ui-test-robot](https://github.com/JetBrains/intellij-ui-test-robot)

Схема взаимодействия

RemoteRobot и IntelliJ IDEA plugin - наш WebDriver / W3C API.



Поиск элементов на экране

1. Поиск компонентов по XPath и атрибутам
2. Исполнение JavaScript

```
projectSteps.createUltimateProject { this: UltimateNewProjectDialogFixture
  setupSpringInitializr { this: SpringInitializrInitialViewFixture
    next.click()
    pause(1000)
    waitForIgnoringError { next.isEnabled() }
    step("Select java 8") {
      find<ComponentFixture>(byXpath("//div[translate(@accessiblename, 'JV', 'jv')='java version:' and @class='ComboBox']"))
        .click()
      find<ComponentFixture>(byXpath("//div[@class='CustomComboPopup' and @name='ComboPopup.popup']"))
        .findText("8").click()
    }
  }
  next.click()
  dependencyDialog { this: DependenciesDialogFixture
    chooseWebStarter()
    assert(selectedDependenciesPanel).hasText("Spring Web")
  }
}
```

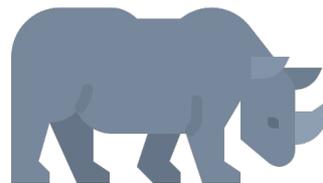
Чем исполнить JavaScript на JVM

- Rhino (Mozilla)  [mozilla/rhino](https://github.com/mozilla/rhino)

- Nashorn Engine (JEP-174)



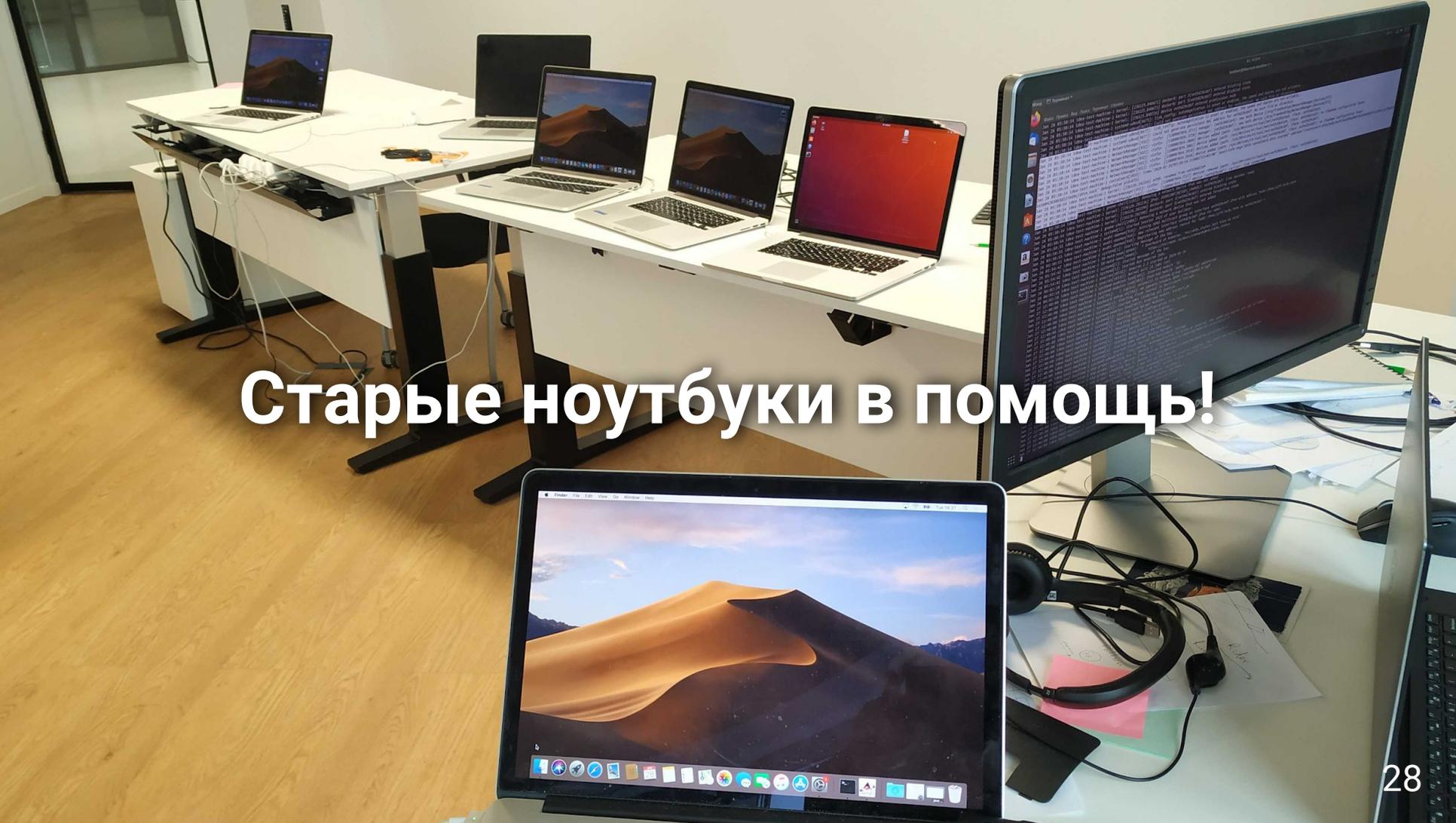
JDK 8 - JDK 15



Удалённое исполнение JavaScript

1. Отправка JavaScript по сети
2. Исполнение JavaScript ES6 при помощи Rhino Engine

```
fun click() = step("click at ${elementData.tag}") {
    scroll()
    container.runJs("""
        robot.click(component, new Point(${x + height / 2}, $centerY))
    """)
}
fun ... .tag}") {
    this
    debugger
    delete ...).elementData
}
do
false
over for
function
```

A photograph of a workstation in a room with light wood flooring. In the background, a row of five old laptops sits on a white desk. The laptops have various desktop backgrounds, including a desert landscape and a red wall. In the foreground, a large monitor displays a terminal window with a dark background and white text, likely showing system logs or code. To the right of the monitor, there are papers, a mouse, and a headset. The overall scene suggests a workspace where older hardware is being used for development or testing.

Старые ноутбуки в помощь!

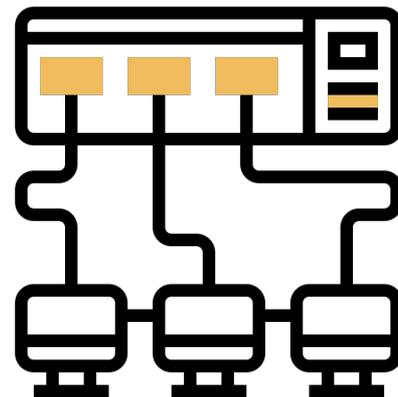
Парк машин

Robot-registry (наш Selenium + Selenium UI) управляет:

1. Реальные машины (старые ноутбуки в серверной)
2. Docker контейнеры с Linux + X11 + XFCE

Возможности:

- Запуск/остановка версий IDE для тестирования
- Получение структуры UI
- Подключение к машинам по VNC
- Резервирование машин для ручной проверки тестов



Robot Registry

 robot-registry

Find container

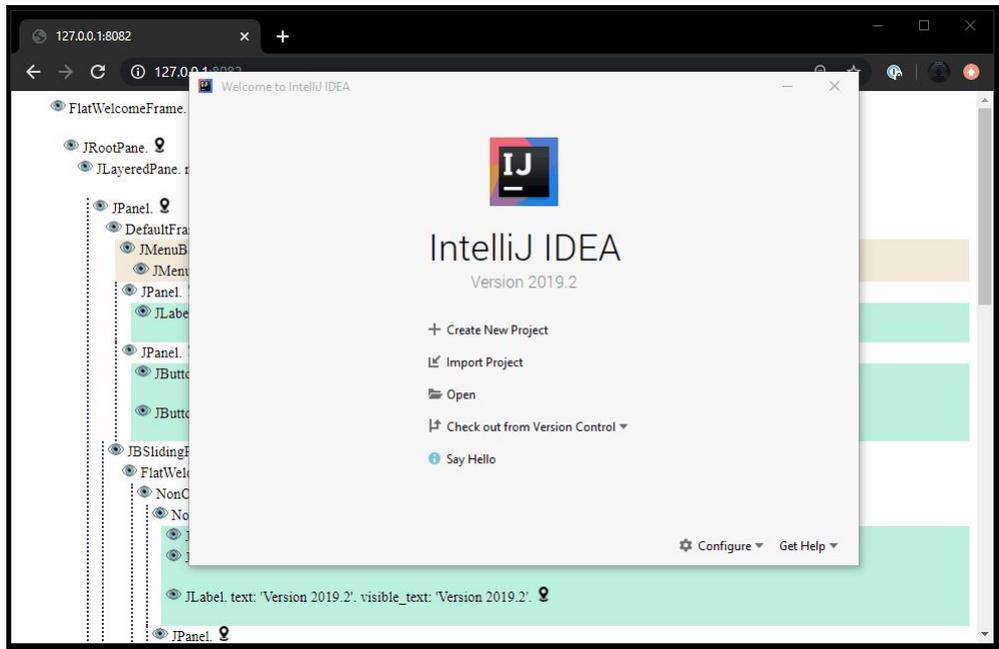
reserve delete more...	idea-img:latest http://172.31.150.195:18182	FavoritesTest	vnc noVnc
reserve delete more...	idea-img:latest http://172.31.144.150:18181	LicenseKeyActivationTest	vnc noVnc
reserve more...	intellij-windows10-ui-1() http://172.31.129.156:8180	KarmaDebugTest. Windows	vnc noVnc
reserve delete more...	idea-img:latest http://172.31.128.151:18182	LicenseKeyActivationTest	vnc noVnc
reserve delete more...	idea-img:latest http://172.31.128.151:18181	FavoritesTest	vnc noVnc

Run ide: [idea-u](#) [202](#) [202.7660.17](#) [launch](#) [Kill ide](#) [status](#) [update 'robot-server'](#) [sync test data](#) [components hierarchy](#) [idea log](#)

For linux: vncviewer 172.31.128.151:15901 &

reserve more...	ubuntu(swift) http://172.30.164.42:8180		
---	---	--	--

Иерархия компонентов Java Swing



ColorBlindnessPanel.

JCheckBox. Visible text: 'Adjust colors for red-green vision deficiency (deuteranopia)'.

ComboBox.

DarculaComboBoxUI.

CellRendererPane.

SwingActionLink. text: 'How it works'. Visible text: 'How it works'.

JCheckBox. Visible text: 'Support screen readers (requires restart)'.

JCheckBox. text: 'Use contrast scrollbars'. Visible text: 'Use contrast'.

JPanel.

JPanel.

JLabel. text: 'Theme:'. Visible text: 'Theme:'.

JComboBox.

DarculaComboBoxUI.

CellRendererPane.

UIResource. text: 'IntelliJ Light'.

JPanel.

FontComboBox.

DarculaComboBoxUI.

CellRendererPane.

FontInfoRenderer.

JLabel. text: 'Size:'. Visible text: 'Size:'.

ComboBox.

DarculaComboBoxUI.

CellRendererPane.

BorderlessTextField. Visible text: '13'.

JCheckBox. text: 'Use custom font:'. Visible text: 'Use custom font:'.

Разбор упавших тестов

1. Сервис отчётов
2. TestRail

The screenshot displays a TestRail report for a failed test case. The test case is titled "Search for License Activation Dialog_error" and is associated with the test run "webstorm-203.4574". The test case is marked as "failed" and has a duration of 73833ms. The test steps are as follows:

- 7:04 8 october 2020: Search 'JDialogFixture' by 'title WebStorm User Agreement' (153ms)
- Search 'Checkbox' by 'accessibleName contains 'I confirm'' (99ms)
- ..click (1024ms)
- Search 'Button' by 'text "Continue"' (21ms)
- ..click (586ms)
- Search 'JDialogFixture' by 'title Data Sharing' (16ms)
- Search 'Button' by 'text "Send Anonymous Statistics"' (15ms)
- ..click (552ms)
- Search for License Activation Dialog (73833ms)

The screenshot of the license activation dialog box shows a window titled "License Activation" with a "Generate License" button and a "Cancel" button. The dialog box is overlaid on a desktop environment.

The right side of the screenshot shows a "history" section with a list of test runs:

- webstorm-203.4449.17: 5:45 8 october 2020, steps passed
- idea-u-203.4571: 2:52 8 october 2020, failed to generate license, steps failed
- webstorm-203.4510: 7:06 7 october 2020, steps passed
- idea-u-203.4507: 2:52 7 october 2020, failed to generate license, steps failed
- idea-u-203.4449.2: 9:49 6 october 2020, steps passed
- webstorm-203.4449: 7:40 6 october 2020, steps passed
- webstorm-203.4449.5: 7:01 6 october 2020, steps passed
- idea-u-203.4446: 2:54 6 october 2020, failed to generate license, steps failed

Визуальные проверки

- Сервис актуальных скриншотов
- Сравнение скриншотов и OCR при помощи aShot:



[pazone/ashot](https://github.com/pazone/ashot)

CheckUltimateUITest

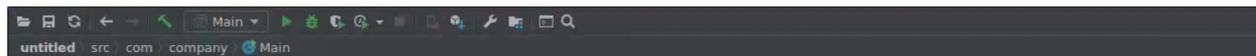
DARCULA Toolbar on

All Linux Windows Mac

Rename

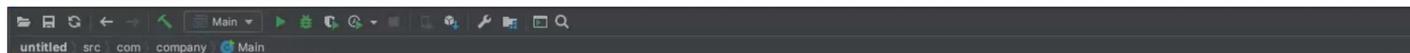
1 OS: linux Created: 9/21/2020, 7:38:23 AM on version: IDEA-U-203.3645.25 Last success: 10/8/2020, 5:56:38 AM use as expected image

Diff with only actual



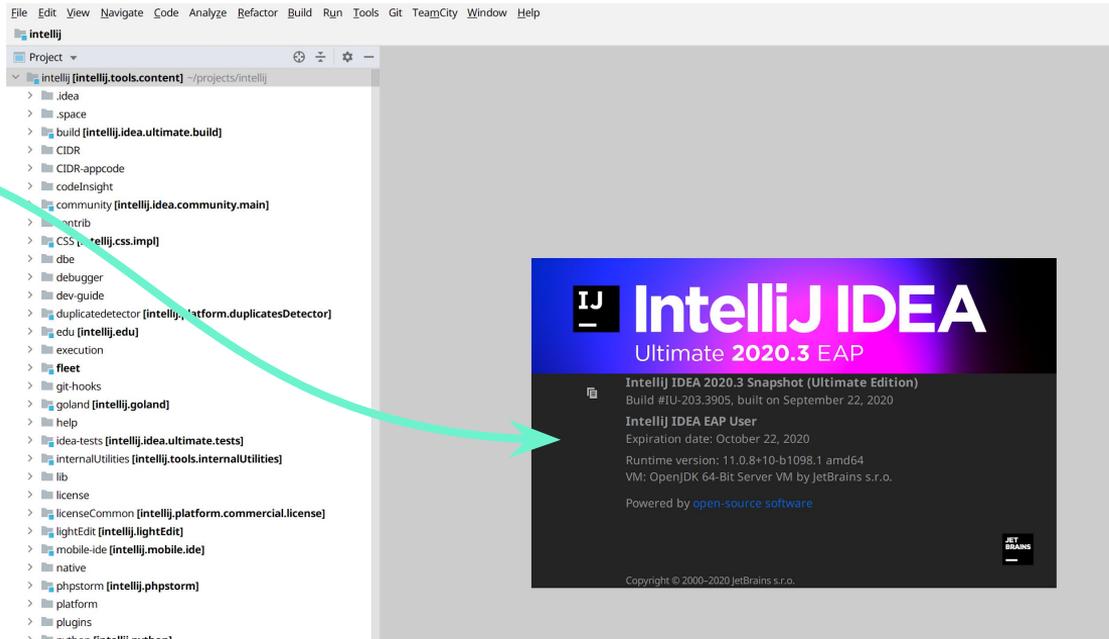
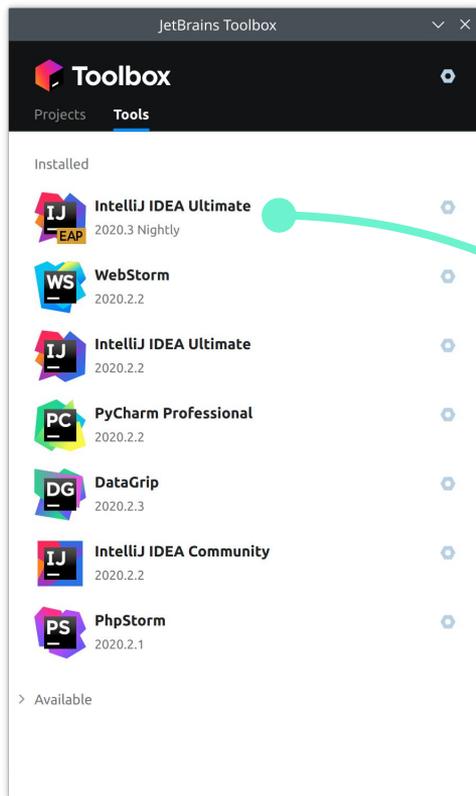
2 OS: mac Created: 9/21/2020, 7:25:22 AM on version: IDEA-U-203.3645.25 Last success: 10/8/2020, 5:49:24 AM use as expected image

Diff with only actual



Как работают команды IDE в JetBrains

Экстремальный догфудинг



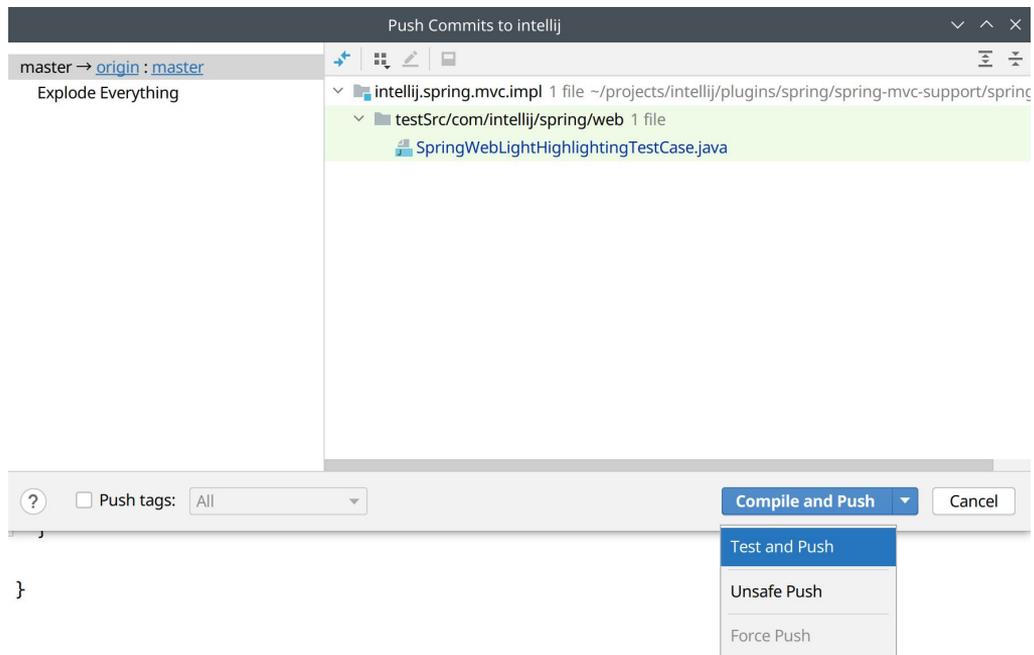
Много автоматизации на CI

- Inspections (статический анализ, SSR, i18n, ...)
- Zero-tolerance Inspections
- Тесты производительности на больших OSS проектах
- Автоматический cherry-pick для веток после ревью
- Автоматическое создание релизных веток
- ...

Safe Push

Как запустить всё в пятницу вечером и потом не сгореть от стыда?

Секретная технология!

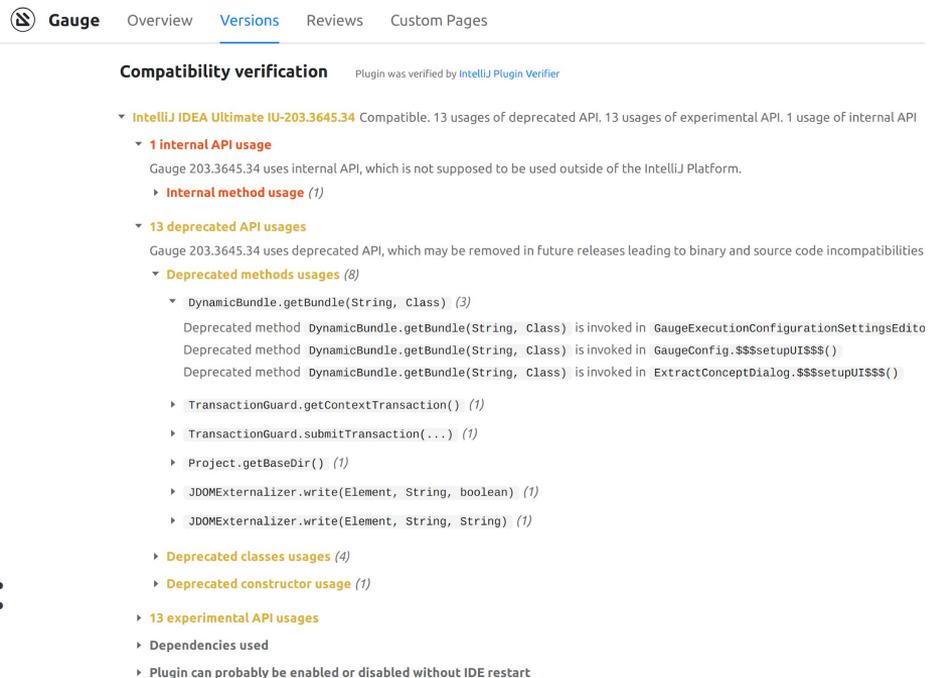


Совместимость API и как это проверить

1. Бинарная совместимость с плагинами
2. Предупреждения об использовании Internal и Deprecated API



* Для Java библиотек воспользуйтесь:



Gauge Overview **Versions** Reviews Custom Pages

Compatibility verification

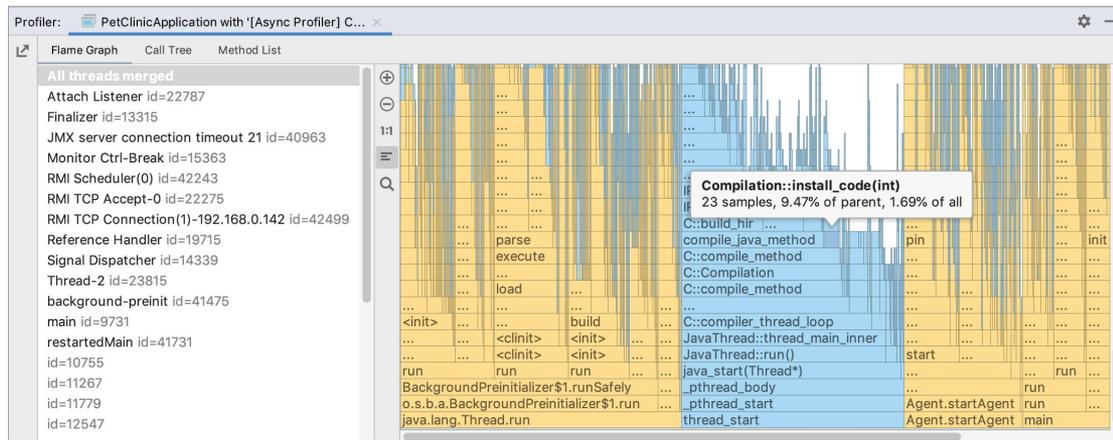
Plugin was verified by [IntelliJ Plugin Verifier](#)

- IntelliJ IDEA Ultimate IU-203.3645.34 Compatible. 13 usages of deprecated API. 13 usages of experimental API. 1 usage of internal API
 - 1 internal API usage**
 - Gauge 203.3645.34 uses internal API, which is not supposed to be used outside of the IntelliJ Platform.
 - Internal method usage** (1)
 - 13 deprecated API usages**
 - Gauge 203.3645.34 uses deprecated API, which may be removed in future releases leading to binary and source code incompatibilities
 - Deprecated methods usages** (8)
 - DynamicBundle.getBundlе(String, Class) (3)
 - Deprecated method DynamicBundle.getBundlе(String, Class) is invoked in GaugeExecutionConfigurationSettingsEdito
 - Deprecated method DynamicBundle.getBundlе(String, Class) is invoked in GaugeConfig.\$\$\$setUpUI\$\$\$()
 - Deprecated method DynamicBundle.getBundlе(String, Class) is invoked in ExtractConceptDialog.\$\$\$setUpUI\$\$\$()
 - TransactionGuard.getContextTransaction() (1)
 - TransactionGuard.submitTransaction(...) (1)
 - Project.getBaseDir() (1)
 - JDOMEexternalizer.write(Element, String, boolean) (1)
 - JDOMEexternalizer.write(Element, String, String) (1)
 - Deprecated classes usages** (4)
 - Deprecated constructor usage** (1)
 - 13 experimental API usages**
 - Dependencies used
 - Plugin can probably be enabled or disabled without IDE restart

Инструментарий анализа производительности

Какими инструментами мы пользуемся для анализа производительности:

1. Debugger (CE)
2. Thread dump analysis (CE)
3. Eclipse MAT (OSS)
4. IDEA + Async Profiler (IU)
5. YourKit (Commercial)



Казуальный профайлер для Java и Kotlin

Spot Profiler:

<https://plugins.jetbrains.com/plugin/13355-spot-profiler-for-java-and-kotlin>

```
44      @Override
45      public void executeWriteAction(Editor editor, Caret caret, DataContext dataContext) {
46          final Project project = editor.getProject();
47          assert project != null;
48          final PsiDocumentManager documentManager = PsiDocumentManager.getInstance(project);
49          final Document document = editor.getDocument();
50          documentManager.commitDocument(document);
51          PsiFile file = getRoot(documentManager.getPsiFile(document), editor);
52
53          if (file != null) {
54              final MoverWrapper mover = getSuitableMover(editor, file); 787ms
55              if (mover != null && mover.getInfo().toMove2 != null) {
56                  LineRange range = mover.getInfo().toMove;
57                  if ((range.startLine > 0 || isDown) && (range.endLine < document.getLineCount() || !isDown)) {
58                      mover.move(editor, file); 552ms
59                  }
60              }
61          }
62      }
```

Куда копать дальше

- Исходный код IntelliJ IDEA CE:
github.com/JetBrains/intellij-community
- Документация Plugin DevKit:
https://www.jetbrains.org/intellij/sdk/docs/basics/testing_plugins/testing_plugins.html
- IntelliJ UI Test Robot:
<https://github.com/JetBrains/intellij-ui-test-robot>

Вопросы ?



Twitter:
[@Yuriy_Artamonov](https://twitter.com/Yuriy_Artamonov)