

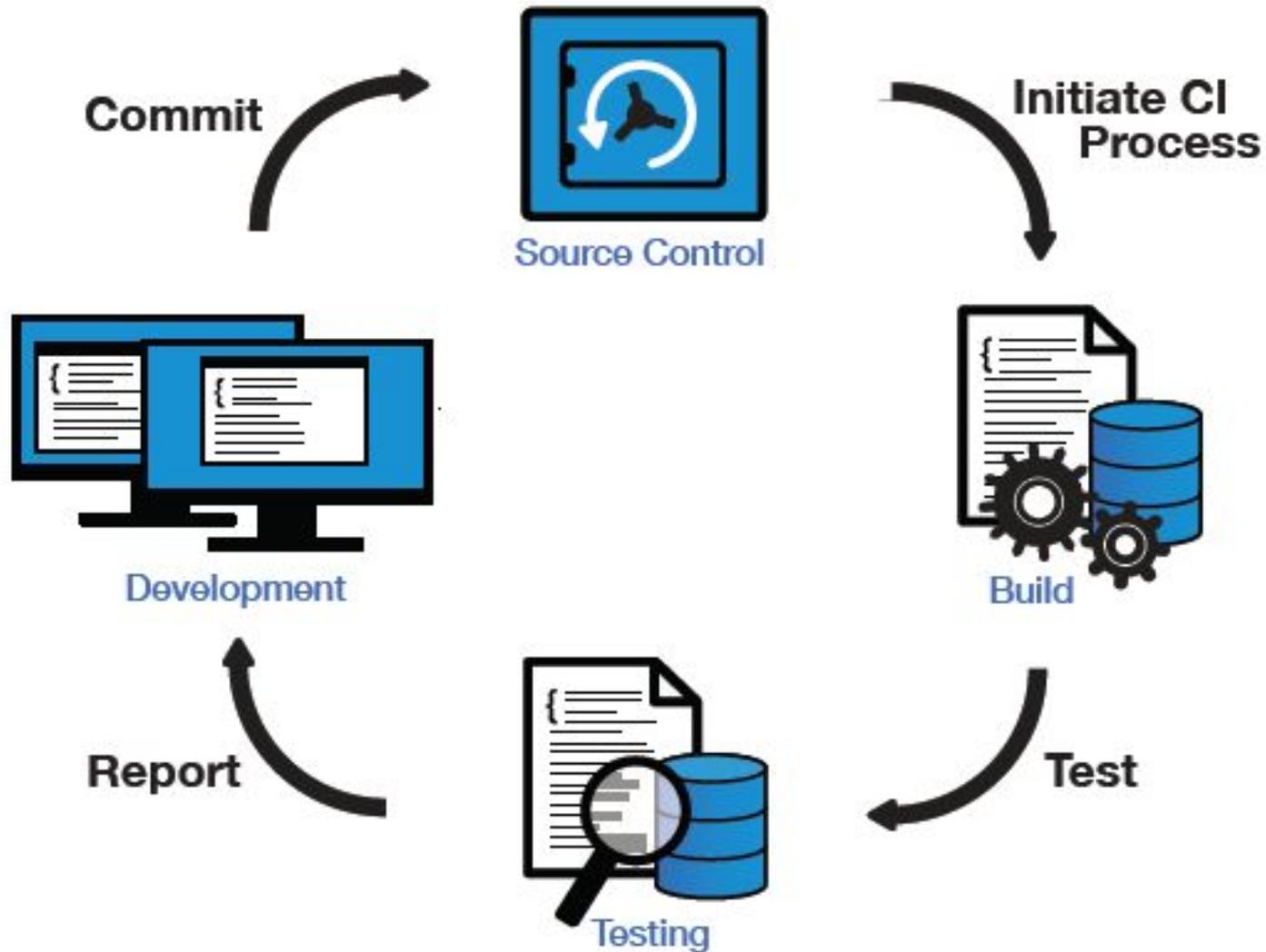
# Continuous Integration

- это практика разработки программного обеспечения, в которой члены команды проводят интеграцию не реже чем раз в день. Результаты интеграции проверяются автоматически, используя автотесты и статический анализ кода.
- Использование CI позволяет вовремя отслеживать ошибки интеграции, сделать систему и процесс разработки более «прозрачными» для всех участников команды
- Фактически, CI позволяет избавиться от предположений, при процессе разработки ПО. Менеджер предполагает, что продукт готов и стабилен, программист — что в коде нет ошибок и т. д.

# Идеологически CI базируется на следующих соглашениях

- часто (не менее 1 раза в день) «заливать» свой код в репозиторий
- писать автоматические тесты
- запускать private builds (процесс сборки, который выполняется с использованием исходного кода, в данный момент находящегося в локальной репозитории разработчика)
- не «заливать» неработающий код
- чинить сломанный build немедленно
- следить за тем, чтобы все тесты и проверки проходили
- не выкачивать из репозитория сломанный код

# Continuous Integration



# Continuous Integration

Базовый процесс интеграции выглядит следующим образом:

- Триггер. Событие, при котором запускается сборка продукта. Таким событием может быть: изменения в коде (push), определенное время, нажатие на кнопку.
- После срабатывания триггера стартует сборка проекта из исходников.
- Развертывание базы данных.
- Развертывание приложения.
- Тесты. Авто-тесты не являются обязательными, но их выполнение крайне желательно. Это один из важных пунктов хороших практик CI.
- Статус, отчеты, уведомления по результатам сборки. После прогона тестов получаем результат сборки, детальные отчеты по каждому из этапов интеграции

# Build script

Скрипт сборки — это набор команд, которые будут выполнены при запуске процесса интеграции. Чаще всего он выглядит как следующий набор шагов:

- Очистка от результатов предыдущего запуска
- Компиляция (или статический анализ кода для интерпретируемых языков)
- Интеграция с базой данных
- Запуск автоматических тестов
- Запуск других проверок (соответствие код стандартам, проверка цикломатической сложности и т. д.)
- Разворачивание программного обеспечения

# Автоматические тесты в СІ

В СІ используются тесты всех уровней за исключением исследовательских:

- модульные (unit) тесты
- компонентные тесты
- функциональные тесты
- системные тесты

По написанию и запуску тестов также принят ряд соглашений:

- более быстрые тесты должны запускаться первыми
- тесты должны быть разделены по категориям
- на все баги должны писаться тесты
- тест кейсы стоит ограничивать одной проверкой
- тесты должны выполняться без ошибок при повторном запуске

# Преимущества CI

- Прежде всего, это регулярная интеграция всего приложения.
- Все делается автоматически, люди избавлены от рутины.
- Экономия времени.
- Работа над проектом прозрачна для всех участников команды. Становится проще ответить на вопросы что? где? когда?
- Уменьшаются риски получить «гранату». Дефекты находятся раньше. Это достигается путем запуска тестов и ранней отдачи нового/измененного функционала на тестирование.
- У нас есть всегда развернутое окружение для тестирования и демонстрации работы заказчику и прочим заинтересованным. Если ваша команда большая, и вы работаете одновременно в разных ветках репозитория. То теперь буквально в несколько движений вы можете настроить ветку кода на нужное окружение или собрать новое с нужной веткой.
- Можно безболезненно эмитировать процесс деплоя на тестовых серверах.
- Хорошая CI система позволяет поддерживать ряд инженерных практик (анализ кода, покрытие кода, юнит-тесты).

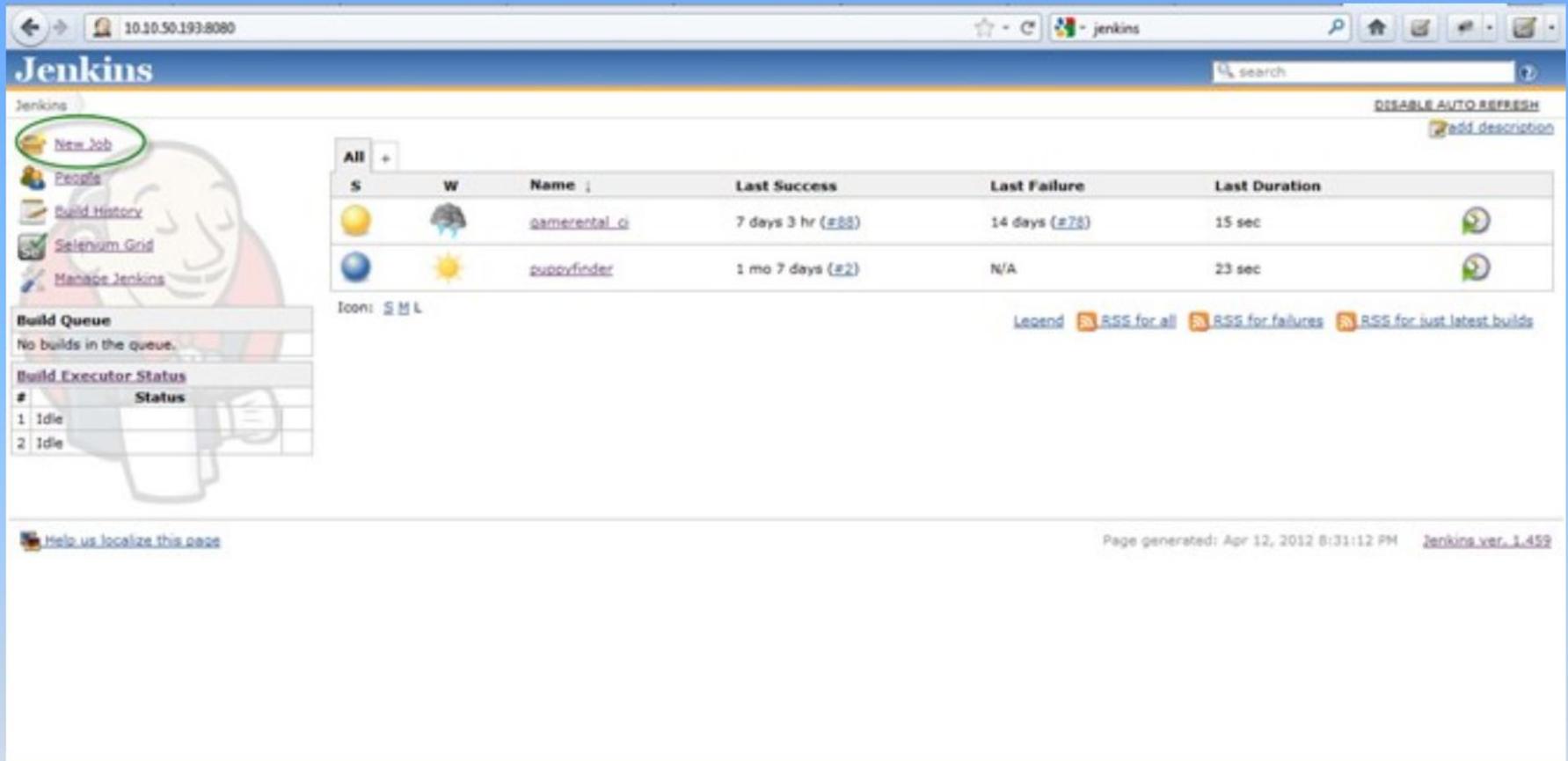
# Узкие места CI

В любом инструменте существует ряд нюансов. Чтобы эффект от использования непрерывной интеграции был наибольшим рекомендуется использовать ряд общепринятых практик.

- Частая синхронизация. В этом заключается основной смысл CI. Нельзя позволять разработчикам длительное время не интегрировать изменения. Хорошей практикой является создание отдельных веток кода и окружений для длительных фич и рефакторинга.
- Решать все проблемы со сборкой максимально быстро. Это позволит избежать простоев. Настройте автоматические нотификации в случае упавшего билда. Это позволит разработчикам своевременно знать о проблемах. Не экономьте на железе. Сервер должен быть мощный.
- Должным образом настройте инфраструктуру. Процесс сборки должен быть надежным.
- Процесс сборки должен быть быстрым, не более 10 минут. Поэтому имеет смысл оптимизировать все шаги сборки. Во время сборки выполняйте только необходимые действия, ничего лишнего. Приемочные тесты не должны занимать много времени, т. к. любые простои не желательны. Пишите и прикручивайте к сборке авто-тесты. Тогда интеграция будет наиболее безопасной.
- CI будет эффективен только тогда, когда вся команды его принимает.
- Расширяйте возможности (анализ покрытия, статические анализаторы, статистика тестов).
- Максимально адаптируйте окружение под production-версию.

# Создание проекта в Jenkins

# Создание проекта (шаг 1)



The screenshot shows the Jenkins web interface. The browser address bar displays '10.10.50.193:8080'. The Jenkins logo is at the top left, and a search bar is at the top right. On the left sidebar, the 'New Job' button is circled in green. Below it are links for 'People', 'Build History', 'Selenium Grid', and 'Manage Jenkins'. The main content area features a table of jobs with columns for status, warning, name, last success, last failure, and last duration. Two jobs are listed: 'qamerental\_ci' and 'suppyfinder'. Below the table are links for 'Legend', 'RSS for all', 'RSS for failures', and 'RSS for just latest builds'. At the bottom, there is a footer with 'Help us localize this page' and 'Page generated: Apr 12, 2012 8:31:12 PM Jenkins ver. 1.459'.

Jenkins

DISABLE AUTO REFRESH  
[add description](#)

All +

S	W	Name ↓	Last Success	Last Failure	Last Duration
		<a href="#">qamerental_ci</a>	7 days 3 hr ( <a href="#">x88</a> )	14 days ( <a href="#">x78</a> )	15 sec
		<a href="#">suppyfinder</a>	1 mo 7 days ( <a href="#">x2</a> )	N/A	23 sec

Icons: [S](#) [M](#) [L](#)

Legend RSS for all RSS for failures RSS for just latest builds

[Help us localize this page](#)

Page generated: Apr 12, 2012 8:31:12 PM Jenkins ver. 1.459

# Создание проекта (шаг 2)

10.10.50.193:8080/view/All/newJob

## Jenkins

Jenkins All

- New Job
- People
- Build History
- Selenium Grid
- Manage Jenkins

**Build Queue**  
No builds in the queue.

**Build Executor Status**

#	Status
1	Idle
2	Idle

Job name: OurNewProject

- Build a free-style software project**  
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be more powerful than software build.
- Build multi-configuration project**  
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Monitor an external job**  
This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed to be a dashboard of your existing automation system. See [the documentation for more details](#).
- Copy existing Job**  
Copy from:

OK

[Help us localize this page](#)

Page generated: Apr 12, 2012

# Создание проекта (шаг 3)

Project name

Description

[Preview](#)

Discard Old Builds

---

### Source Code Management

CVS

Git

Repositories

Branches to build

Repository browser

None

**Server returned HTTP response code: 502 for URL: http://code.quartsoft.com/browse/GameRental/**

# Создание проекта (шаг 4)

**Build**

☰ **Invoke Phing targets**

Targets

Phing Build file

Properties

Use ModuleRoot as working directory

# Создание проекта (шаг 5)

**Post-build Actions**

Publish Checkstyle analysis results

Publish PMD analysis results

Publish duplicate code analysis results

Duplicate code results

[Fileset includes](#) setting that specifies the generated raw XML report files, such as `**/cpd.xml` or `**/simian.xml`. Basedir of the fileset is [the workspace root](#). If no value is set, then the default `**/cpd.xml` is used. Be sure not to include any non-report files into this pattern.

High priority threshold

Minimum number of duplicated lines for high priority warnings.

Normal priority threshold

Minimum number of duplicated lines for normal priority warnings.

Aggregate downstream test results

Archive the artifacts

Build other projects

Plot build data

Publish Clover PHP Coverage Report

Clover XML location

Specify the name of the Clover xml file generated relative to [the workspace root](#).

Publish HTML Report

Clover HTML report directory

Specify the path to the directory that contains the Clover HTML report file, relative to [the workspace root](#)

Disable report archiving

Publish JUnit test result report

Test report XMLs

[Fileset includes](#) setting that specifies the generated raw XML report files, such as `*myproject/target/test-reports/*.xml`. Basedir of the fileset is [the workspace root](#).

Retain long standard output/error

# Запускаем билд:

**Jenkins**  
Jenkins > gamerental\_ci

**Project gamerental\_ci**

[Back to Dashboard](#)  
[Status](#)  
[Changes](#)  
[Workspace](#)  
**[Build Now](#)**  
[Delete Project](#)  
[Configure](#)  
[Duplicate Code](#)  
[Clover HTML Report](#)

[Clover HTML Report](#)  
[Workspace](#)  
[Recent Changes](#)  
[Latest Test Result \(2 failures / -4\)](#)

**Build History** (trend)

#89	<a href="#">Apr 12, 2012 8:55:22 PM</a>
#88	<a href="#">Apr 5, 2012 5:23:26 PM</a>
#87	<a href="#">Apr 4, 2012 7:21:10 PM</a>
#86	<a href="#">Apr 3, 2012 9:53:52 PM</a>
#85	<a href="#">Mar 29, 2012 8:44:48 PM</a>
#84	<a href="#">Mar 29, 2012 8:37:25 PM</a>
#83	<a href="#">Mar 29, 2012 8:09:35 PM</a>
#82	<a href="#">Mar 29, 2012 8:00:10 PM</a>
#81	<a href="#">Mar 29, 2012 7:50:49 PM</a>
#80	<a href="#">Mar 29, 2012 7:49:27 PM</a>
#79	<a href="#">Mar 29, 2012 6:31:04 PM</a>
#78	<a href="#">Mar 29, 2012 6:30:09 PM</a>
#77	<a href="#">Mar 29, 2012 6:10:42 PM</a>
#76	<a href="#">Mar 29, 2012 6:07:41 PM</a>
#75	<a href="#">Mar 29, 2012 2:12:06 PM</a>
#74	<a href="#">Mar 28, 2012 8:47:19 PM</a>
#73	<a href="#">Mar 28, 2012 7:43:07 PM</a>

[RSS for all](#) [RSS for failures](#)

**Permalinks**

- [Last build \(#89\), 7 days 3 hr ago](#)
- [Last stable build \(#84\), 14 days ago](#)
- [Last successful build \(#88\), 7 days 3 hr ago](#)
- [Last failed build \(#78\), 14 days ago](#)
- [Last unstable build \(#88\), 7 days 3 hr ago](#)
- [Last unsuccessful build \(#88\), 7 days 3 hr ago](#)

# Анализируем результат (ВЫВОД ИЗ КОНСОЛИ):



The screenshot shows the Jenkins web interface for a build named 'game rental\_ci'. The main area displays the 'Console Output' for a build started by user 'anonymous'. The output shows the build process, including fetching changes from a Git repository, checking out a specific revision, and running a series of phing targets. The targets include creating directories, deleting old ones, and running various scripts like 'ylic', 'phpunit', and 'phpcpd'. The build concludes with 'BUILD FINISHED' and a total time of 25.8188 seconds.

**Executed Phing Targets**

- update
- phpcpd
- phpunit
- build

```
Started by user anonymous
Building in workspace /var/lib/jenkins/jobs/game rental_ci/workspace
Checkout:workspace / /var/lib/jenkins/jobs/game rental_ci/workspace - Hudson.remoting.LocalChannel@0f394b
Using strategy: Default
Last Built Revision: Revision 1e0453e60ec28d22b30366527439ff2c74e1e62 (origin/testing)
Fetching changes from 1 remote Git repository
Fetching upstream changes from git@pm.quart-soft.com:game rental
Commencing build of Revision 1e0453e60ec28d22b30366527439ff2c74e1e62 (origin/testing)
Checking out Revision 1e0453e60ec28d22b30366527439ff2c74e1e62 (origin/testing)
looking for "/var/lib/jenkins/jobs/game rental_ci/workspace/phing_project.xml" ...
use "/var/lib/jenkins/jobs/game rental_ci/workspace" as a working directory.
[workspace] $ phing -buildfile /var/lib/jenkins/jobs/game rental_ci/workspace/phing_project.xml build -logger phing.listener.DefaultLogger
Buildfile: /var/lib/jenkins/jobs/game rental_ci/workspace/phing_project.xml

Game rental > update:

    [echo] Making directory ./assets ./protected/runtime ./protected/views/emails/compiled
[delete] Deleting directory /var/lib/jenkins/jobs/game rental_ci/workspace/assets
[mkdir] Created dir: /var/lib/jenkins/jobs/game rental_ci/workspace/assets
[delete] Deleting directory /var/lib/jenkins/jobs/game rental_ci/workspace/protected/runtime
[mkdir] Created dir: /var/lib/jenkins/jobs/game rental_ci/workspace/protected/runtime
[delete] Deleting directory /var/lib/jenkins/jobs/game rental_ci/workspace/protected/views/emails/compiled
[mkdir] Created dir: /var/lib/jenkins/jobs/game rental_ci/workspace/protected/views/emails/compiled
[echo] done...
[echo] Making ylic executable
[chmod] Changed file mode on "/var/lib/jenkins/jobs/game rental_ci/workspace/protected/ylic" to "755"
[echo] done...
[echo] Starting DB migration
[echo] done...
[echo] Making directory for phpunit logs
[delete] Deleting directory /var/lib/jenkins/jobs/game rental_ci/workspace/build/logs
[mkdir] Created dir: /var/lib/jenkins/jobs/game rental_ci/workspace/build/logs
[delete] Deleting directory /var/lib/jenkins/jobs/game rental_ci/workspace/build/coverage
[mkdir] Created dir: /var/lib/jenkins/jobs/game rental_ci/workspace/build/coverage
[echo] done...

Game rental > phpcpd:

    [echo] Find duplicate code using PHPCPD
[echo] done...

Game rental > phpunit:

    [echo] Run tests using PHPUnit
[echo] done...

Game rental > build:

    [echo] The end.

BUILD FINISHED

Total time: 25.8188 seconds
```

# Анализируем результат (общая информация):

The screenshot shows the Jenkins web interface for a specific build. The browser address bar indicates the URL is `10.10.50.193:8080/job/gamerental_ci/88/`. The page title is "Jenkins". The breadcrumb navigation shows "Jenkins > gamerental\_ci > #88".

The main content area displays the build details for "Build #88 (Apr 5, 2012 5:23:26 PM)". The build status is indicated by a yellow circle icon.

On the left side, there is a navigation menu with the following items:

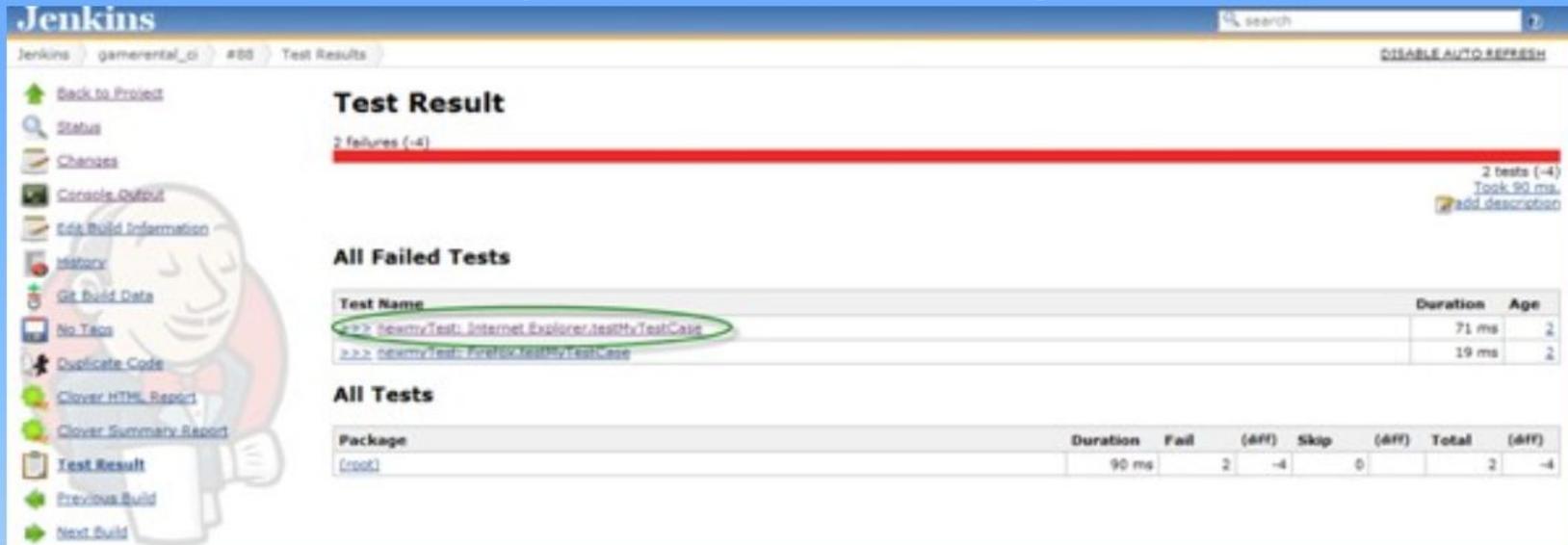
- [Back to Project](#)
- [Status](#)
- [Changes](#)
- [Console Output](#)
- [Edit Build Information](#)
- [Git Build Data](#)
- [No Tags](#)
- [Duplicate Code](#)
- [Clover HTML Report](#)
- [Clover Summary Report](#)
- [Test Result](#)
- [Previous Build](#)
- [Next Build](#)

The main content area contains the following information:

- Changes** (circled in green):
  1. new phpwebdriver added ([detail](#) / [FishEye](#))
  2. new phpwebdriver added ([detail](#) / [FishEye](#))
  3. new phpwebdriver added ([detail](#) / [FishEye](#))
- Started by anonymous user**
- Revision:** 1e0653e60ec28d822b30366527439ff2c74e1e62
  - origin/testing
- Duplicate Code:** [20 warnings](#) from one analysis.
- Clover Code Coverage** - 0% method 0/0, statement 0/0.
- Test Result** (2 failures / -4) ([Show all failed tests >>>](#)) (circled in green)

At the bottom left, there is a link: [Help us localize this page](#). At the bottom right, it says: "Page generated: Apr 3".

# Анализируем результат (Test Results):

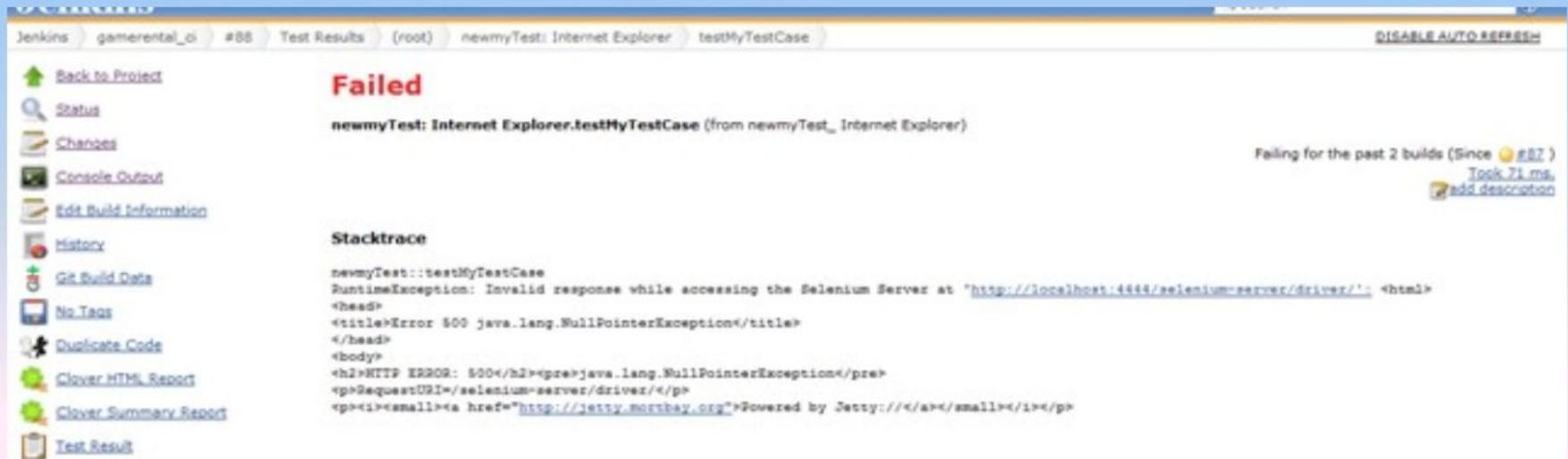


The screenshot shows the Jenkins Test Results page for a build named 'game rental\_o' with ID '#88'. The page title is 'Test Result' and it indicates '2 failures (-4)'. A red bar at the top highlights the failure status. On the left, there is a navigation menu with options like 'Back to Project', 'Status', 'Changes', 'Console Output', 'Edit Build Information', 'History', 'Git Build Data', 'No Tags', 'Duplicate Code', 'Clover HTML Report', 'Clover Summary Report', 'Test Result', 'Previous Build', and 'Next Build'. The main content area shows 'All Failed Tests' with a table listing two failed tests:

Test Name	Duration	Age
<a href="#">newmyTest: Internet Explorer.testMyTestCase</a>	71 ms	2
<a href="#">newmyTest: Firefox.testMyTestCase</a>	19 ms	2

Below this, the 'All Tests' section shows a summary table:

Package	Duration	Fail	(diff)	Skip	(diff)	Total	(diff)
[root]	90 ms	2	-4	0		2	-4



The screenshot shows the Jenkins Test Results page for a specific failed test case: 'newmyTest: Internet Explorer.testMyTestCase (from newmyTest\_ Internet Explorer)'. The page title is 'Failed' and it indicates 'Failing for the past 2 builds (Since #87)'. The main content area shows the 'Stacktrace' for the failed test:

```
newmyTest::testMyTestCase
RuntimeException: Invalid response while accessing the Selenium Server at "http://localhost:4444/selenium-server/driver/": <html>
<head>
<title>Error 500 java.lang.NullPointerException</title>
</head>
<body>
<h1>HTTP ERROR: 500</h1><pre>java.lang.NullPointerException</pre>
<p>RequestURI=/selenium-server/driver/</p>
<p><small><a href="http://jetty.mortbay.org">Powered by Jetty://</a></small></p>
```

# Создание проекта в TeamCity

# Создаем новый проект

Administration > Create New Project

Parent Project: \*

Name: \*

Project ID: \*

This ID is used in URLs, REST API, HTTP requests to the server, and configuration settings in the TeamCity Data Directory.

Description:

## Further steps

You need to create a project first and then proceed to configure its VCS settings and build configurations.

Create

Cancel

# Создание «build configuration» (конфигурации сборки)



Projects ▾

My Changes

Agents 1

Build Queue 0

Administration > <Root project> > test project > Create Build Configuration

## General Settings

Name: \*

Build configuration ID: \*



This ID is used in URLs, REST API, HTTP requests to the server, and configuration settings in the TeamCity Data Directory.

Description:

# Добавление нового репозитория

**TC** Projects | My Changes | Agents 1 | Build Queue 0

Administration > <Root project> > test project > Create Build Configuration

## Version Control Settings

There are no VCS roots attached to this build configuration.

[+ Create and attach new VCS root](#)

---

### Checkout Settings

**VCS checkout mode:**

**Checkout directory:**

With this selection all build configurations with the same VCS settings will use the same checkout directory.

---

**Clean all files before build:**

---

### Display settings

**Display options:**  Show changes from snapshot dependencies

---

[<< General Settings](#) [Add Build Step >>](#) [Cancel](#)

# Добавление нового репозитория



Projects ▾

My Changes

Agents 90



Build Queue 69

Administration > Edit Build Configuration > Edit VCS Root

## Type of VCS

Type of VCS:

Git

## VCS Root Name and ID

VCS root name: \* <sup>?</sup>

https://git.example.com/project.git

Enter a unique name to distinguish this VCS root from other roots. If not specified, the name will be generated automatically.

VCS root ID: \* <sup>?</sup>

ID

Regenerate

VCS root ID must be unique across all VCS roots. VCS root ID can be used in parameter references to VCS root parameters and REST API.

## General Settings

Fetch URL: \*

https://git.example.com/project.git

It is used for fetching data from repository.

Push URL:

It is used for pushing tags to the remote repository. If blank, the fetch url is used.

Default branch: \*

dev

Branch to be used if no branch from Branch Specification is set

# Добавление нового репозитория

Authentication Settings

Authentication method: Password

User name: user.logn

Username must be specified if there is no username in the clone URL. The user name specified here overrides username from URL.

Password: .....

**Authentication method** – способ аутентификации. Тут может быть и доступ без авторизации (если данные лежат на внутреннем сервере, например), и по ключу, и по паролю. Для каждого варианта дополнительные поля будут свои

# Настройка автоматического запуска задания

**Add New Build Trigger** ×

VCS Trigger ▼

*VCS Trigger will add a build to the queue when a VCS check-in is detected.*

---

Trigger on changes in snapshot dependencies

---

Per-checkin Triggering

Trigger a build on each check-in

Include several check-ins in a build if they are from the same committer

---

Quiet Period Settings

**Quiet period mode:** <sup>?</sup>

Do not use

Use default value (60 seconds)

Custom  seconds

---

VCS Trigger Rules

**Trigger rules:** <sup>?</sup> [Edit Trigger Rules](#)

Newline-delimited set of rules: +|-:[Ant-like wildcard]

---

# Создание шагов сборки

 [Projects](#) | [My Changes](#) | [Agents](#) 1  | [Build Queue](#) 0 | [admin](#) | [Administration](#)

Administration >  <Root project> >  test project >  Auto: build and test #dev

[Run](#) ... | [Build Configuration Home](#)

## Build Steps

[+ Add build step](#)

## Additional Build Features

There are no build features configured.

[+ Add build feature](#)

## Configuration Steps

- 1 [General Settings](#)
- 2 [Version Control Settings](#)
- 3 [Build Step: MSBuild](#)
- 4 [Build Failure Conditions](#)

# Создание шагов сборки



Projects ▾

My Changes

Agents 1

Build Queue 0

Administration > <Root project> > test  
project > Create Build Configuration

## New Build Step

Runner type:

-- Choose build runner type -- ▾

<< VCS settings

Save

Cancel



Administration > <Root project> > test  
project > Create Build Configuration

## New Build Step

Runner type:

MSBuild

Runner for MSBuild files

Step name:

Сборка приложения.

Optional, specify to distinguish this build step from other steps.

Execute step:

If all previous steps finished successfully (zero exit code)

Specify the step execution policy.

Build file path: \*

example\example.sln

The specified path should be relative to the checkout directory.

Working directory:

Optional, set if differs from the checkout directory.

MSBuild version:

Microsoft .NET Framework 4.5

MSBuild ToolsVersion:

4.0

Run platform:

x86

Targets:

Enter targets separated by space or semicolon.

Command line  
parameters:

/p:Configuration=Local

Enter additional command line parameters to MSBuild.exe.

Reduce test failure  
feedback time:

Run recently failed tests first

.NET Coverage

.NET Coverage tool:

<No .NET Coverage>

Choose a .NET coverage tool.

Test code coverage is supported only for NUnit tests run using TeamCity facilities.

<< VCS settings

Save

Cancel

# Добавление параметров сборки

Administration > <Root project> > test project > Auto: build and test #dev

Run ... | Build Configuration Home

+ Add new parameter

Configuration Parameters <sup>?</sup>  
None defined

System Properties (system.) <sup>?</sup>  
None defined

Environment Variables (env.) <sup>?</sup>  
None defined

### Add New Parameter

**Name:** \*

**Kind:**

**Value:**   
Type '%' for reference completion

**Spec:**   
[Show raw value](#)  
Defines parameter control presentation and validation.

Configuration Steps

- 1 General Settings
- 2 Version Control Settings
- 3 Build Steps (4)
- 4 Build Failure Conditions
- 5 Build Triggers (1)
- 6 Dependencies
- 7 Build Parameters

# Общий вид прошедших сборок



Projects ▾

Changes

Agents 30



Build Queue 24

Automation

## Run E2E Smoke Tests - PREPARE JSON DATA

Overview

History

Change Log

Issue Log

Statistics

Compatible Agents 18

Pending Changes 63

Settings

### Pending changes

63 changes ▾

### Current status

Idle

### Recent history

Filter by tag: **master** staging

	Results	Artifacts	Changes	Started	Duration
#20	✓ Success ▾	View ▾	Changes (4) ▾	23 Jun 17 08:48	1m:03s
#19	✓ Success ▾	View ▾	Changes (8) ▾	22 Jun 17 08:58	1m:04s
#18	✓ Success ▾	View ▾	No changes ▾	19 Jun 17 16:22	1m:12s
#17	✓ Success ▾	View ▾	No changes ▾	15 Jun 17 09:16	58s
#16	✓ Success ▾	View ▾	No changes ▾	15 Jun 17 09:09	1m:25s
#15	✓ Success ▾	View ▾	No changes (1) ▾	15 Jun 17 09:04	1m:07s
#14	✓ Success ▾	View ▾	Changes (3) ▾	14 Jun 17 14:38	1m:05s
#13	✓ Success ▾	None ▾	1 (1) ▾	09 Jun 17 13:44	1m
#12	✓ Success ▾	None ▾	(1) ▾	09 Jun 17 13:24	1m
#11	✓ Success ▾	None ▾	(1) ▾	09 Jun 17 11:52	59s
#10	✓ Success ▾	None ▾	No changes ▾	08 Jun 17 14:36	1m:05s
#9	✓ Success ▾	None ▾	No changes ▾	08 Jun 17 14:26	1m:02s

# Детальный лог сборки



Projects | ▾

Changes

Agents 29



Build Queue 18

Automation / Run E2E Smoke Tests - PREPARE JSON DATA

✓ #20 (23 Jun 17 08:48) | ▾

Overview

Changes 4

Build Log

Parameters

Artifacts

Tree view | Tail



View: All messages ▾

Console view

```
[08:48:13] The build is removed from the queue to be prepared for the start
[08:48:13] [E] Collecting changes in 5 VCS roots
[08:48:14] Starting the build on the agent TPLTCRH7-555898
[08:48:15] Clearing temporary directory: /opt/teamcity-agent/temp/buildTmp
[08:48:15] [E] Publishing internal artifacts
[08:48:15] Will perform clean checkout. Reason: Checkout directory is empty or doesn't exist
[08:48:15] Checkout directory: /opt/teamcity-agent/work/1c7a6ea002bfa93f
[08:48:15] [E] Updating sources: server side checkout (12s)
[08:48:28] [E] Step 1/3: Prepare server, install dependencies (Command Line) (46s)
[08:49:14] [E] Step 2/3: Collect all e2e test in one folder (Command Line)
[08:49:14] [E] Step 3/3: Prepare JSON-data files (Command Line) (3s)
[08:49:18] [E] Publishing internal artifacts
[08:49:18] [E] Publishing artifacts
[08:49:18] Build finished
```