МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования «Саратовский государственный

технический университет имени Гагарина Ю.А.»

РЕФЕРАТ

на тему «Система непрерывной интеграции TeamCity»

Выполнила:

Студентка группы мИВЧТ-21

Харьковская Н.В.

Проверил:

Кинцель Д.А.

Саратов 2013

Содержание

[1. Описание TeamCity. 2](#_Toc345581042)

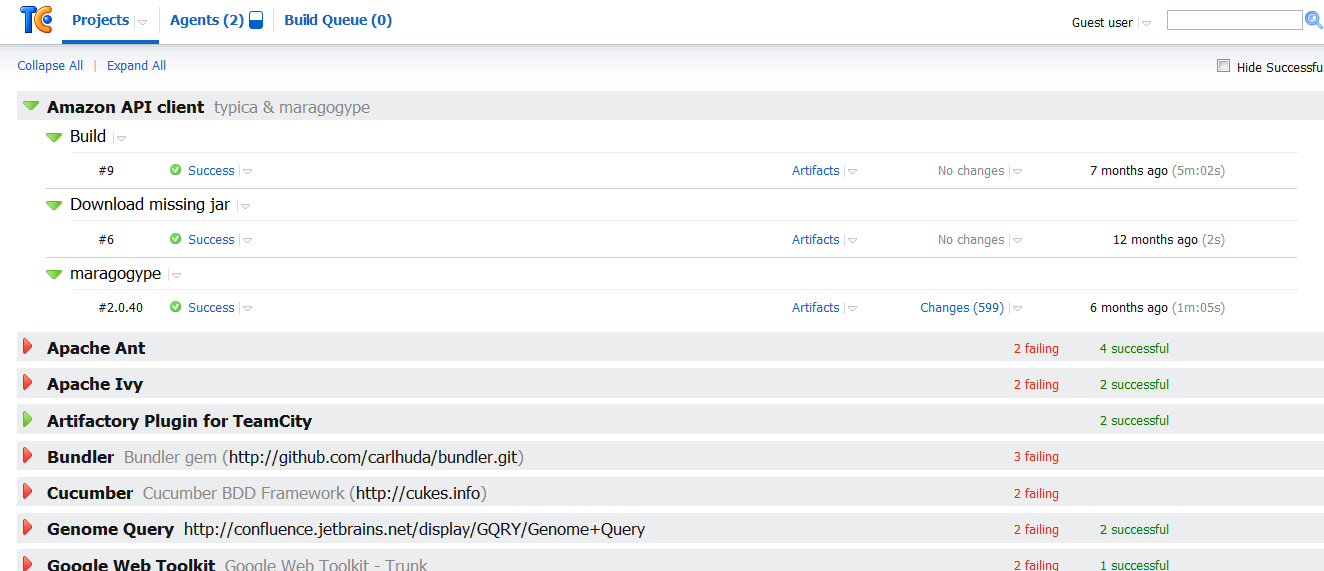
[2. Натсройка Teamcity. 14](#_Toc345581043)

[3. Список используемых источников 24](#_Toc345581044)

1. Описание TeamCity.



**JetBrains TeamCity** - это система непрерывной интеграции и управления разработками, для организации действенной коллективной работы над кодом приложений Java и .NET. TeamCity идет в пакете с плагинами для Eclipse, Microsoft Visual Studio, и IntelliJ IDEA, которые обеспечивают продуктивные способности для интегрированных сред разработки. TeamCity автоматизирует и координирует ключевые совместные процессы, обеспечивает тесную интеграцию с многочисленными средствами разработки, а также ускоряет и упрощает процессы тестирования и анализа кода приложений (скачать: <http://www.jetbrains.com/teamcity/download/index.html>).

[](javascript:ShowImg('/images/upload/1/39f77f71efb5ae3814717d1fba9dadde.png','1324','571');)

**Лицензия**

Возможны следующие варианты лицензирования:

* Professional Server License: Бесплатная версия. Имеется ограничение в 20 конфигураций сборки build configurations; Включает в себя лицензию на 3 [агента](http://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D1%8B%D0%B9_%D0%B0%D0%B3%D0%B5%D0%BD%D1%82) для сборки.
* Enterprise Server License: Коммерческая версия. Неограниченное количество пользовательских аккаунтов и конфигураций сборки; Включает в себя лицензию на 3 [агента](http://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D1%8B%D0%B9_%D0%B0%D0%B3%D0%B5%D0%BD%D1%82) для сборки.
* Build Agent License: Коммерческая, позволяет добавить дополнительно ещё один [агента](http://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D1%8B%D0%B9_%D0%B0%D0%B3%D0%B5%D0%BD%D1%82) для сборки к уже имеющимся. Может использоваться совместно как с лицензией Professional Server License, так и с лицензией and Enterprise Server License.
* Open Source License.

**Интеграция и тестирование.**

TeamCity автоматизирует процесс выполнения тестов и внесения изменений в код членами группы. TeamCity позволяет проводить комплексное тестирование, присылает уведомления об обнаруженных ошибках, и обеспечивает выполнение рутинных операций по интеграции. Всё что нужно сделать разработчику – внести изменения в систему контроля версий и продолжать работать.

Возможность проводить тестирование удалённо с помощью инструмента Build Grid, который позволяет задействовать несколько компьютеров сети. Отчёты о не пройденных тестах мгновенно предоставляются членам группы, которым необходима эта информация, а результаты тестирования доступны всей группе в режиме он-лайн.

**Эффективное управление блоками кода.**

TeamCity обеспечивает гибкость управления блоками кода. Можно определять и запускать различные типы блоков (ежедневный, периодический и др.) с различными конфигурациям для любого проекта, включая настраиваемые триггеры, обеспечивающие удобство и гибкость управления процессом создания кода.

**Своевременное и автоматизированное сообщение внутри группы.**

Настраиваемая таблица уведомлений обеспечивает немедленное уведомление разработчика об ошибке, вызванной его кодом, и отображение ошибки через веб-интерфейс. Одним щелчком мыши разработчик может не только взяться за исправление ошибки, но и проинформировать всю группу о том, что кто-то занимается исправлением (а также сообщить, когда исправление будет выполнено).

**Анализ на стороне сервера.**

Помимо возможности проведения тестов во время создания блоков, для блоков, имеющих конфигурацию, как файл проекта IntelliJ IDEA, можно удалённо проводить набор инспекций кода IntelliJ IDEA. Все обнаруженные проблемы будут отображаться в веб-интерфейсе и редакторе IntelliJ IDEA так же, как при локальных инспекциях.

**Веб-интерфейс и инструментальная панель проекта.**

Веб-браузер – это всё, что нужно для работы в TeamCity, при этом не привязаны к какой-либо среде разработки или языку программирования. Функциональный веб-интерфейс позволяет выполнять все административные и повседневные задания. Панель инструментов обеспечивает просмотр текущих блоков и неуспешно выполненных тестов, а также запуск блоков и реорганизацию очереди блоков.

**Интеграция со средой разработки.**

Специальный модуль обеспечивает интеграцию между TeamCity и IntelliJ IDEA 6.0, давая возможность разработчикам воспользоваться функциями TeamCity в среде разработки.

**Функция предварительного тестирования.**

Эта уникальная функция модуля TeamCity для IntelliJ IDEA позволяет избежать часто встречающегося «синдрома поздней проверки». Перед внесением изменений в код, они подвергаются тестированию. Если тест выполнен успешно, TeamCity автоматически вносит изменения в систему контроля версий. Оттуда они автоматически будут интегрированы в следующий блок TeamCity. Если тест выполнен неуспешно, код не принимается, и его разработчик получает уведомление. Код с дефектом никогда не попадает в блок, таким образом, не нарушается работа всей группы.

**Удалённый запуск личных блоков.**

Данная функция модуля TeamCity позволяет избежать печальных экспериментов с кодом и практически устраняет возможность внесения разработчиком кода, который может вызвать ошибку в блоке. Разработчики могут запускать личные блоки удалённо с помощью TeamCity Build Grid, как обычные блоки, но при этом изменённый код не вносится в систему контроля версий. Если происходит ошибка, «реальный» блок не пострадает, таким образом, не создаётся помех работе группы. Получив мгновенное уведомление об обнаруженных ошибках, разработчик может заняться их исправлением.

Когда изменённый код проходит проверку в личном блоке, разработчик может с уверенностью внести его в реальный блок, зная, что он не вызовет ошибку. Таким образом, разработчики могут свободно экспериментировать, не создавая при этом проблем для общего рабочего процесса.

**Самый быстрый отклик системы.**

Настройка изменения очерёдности тестов, отчёты о результатах в режиме реального времени, настраиваемые уведомления и даже возможность делать образцы разработок доступными до полного завершения самих разработок - **TeamCity** информирует о последних обновлениях разработок и промежуточных результатов, а также показывает, насколько хорошо изменения интегрируются в проект.

**Широкая универсальность - благодаря "облачным" вычислениям.**

Предлагаемое управление разработками помогает оптимизировать утилизацию ресурсов Вашего компьютера при помощи синхронизирования разработок продукта внутри формата агентов разработки, а интеграция с удалённым хранилищем данных Amazon EC2 предлагает простой и эффективный способ универсализации этого формата разработки для осуществления различных запросов случайно отобранными блоками пробных решений.

**Предварительная проверка качества**

TeamCity автоматизирует более 600 инвентаризаций Java-кода, областей кода и осуществляет поиск по заданному образцу - и это является инновацией программы. Также она создаёт, проверяет и запускает автоматические тесты на сервере даже до фиксирования изменений - поддерживая порядок в базе кодов на всех этапах.

**Поддержка Maven**

Создание конфигураций разработок, основанных на Maven (нужна только ссылка на файл POM), визуализация различных проектных данных, собранных из POM, а также новый тип пусковой команды зависимости - пусковая команда Maven. Этот тип пусковой команды запускает разработку каждый раз, когда меняется отдельный компонент Maven.

**Шаблоны конфигурации разработки**

Шаблоны могут спасти от большого количества ручной работы, когда нужно описать некоторое количество одинаковых или очень похожих конфигураций разработок. Можно создать шаблон со стандартными (общими) настройками, а затем скопировать из этого шаблона любое количество конфигураций разработок. А также можете извлечь шаблон из любой уже существующих конфигураций разработок.

**Интеграция трекера событий**

Интеграция трекера событий предлагает фиксированный заголовок ошибки, прямую навигацию по событию и другие улучшения. В данный момент она доступна для Jira, Bugzilla и JetBrains YouTrack отдельно от общего пакета, но может быть встроена в другие системы при помощи плагинов.

**Средство командной строки для персональных разработок**

Раньше, если не использовалась Eclipse, IntelliJ IDEA или Visual Studio, то не было возможности использовать одну из самых важных функций **TeamCity** - удалённый запуск и осуществление предварительных тестов. На данный момент можно осуществлять разработку локальных изменений удалённо с сервера - из локальной командной строки. Не требуется никаких интегрированных сред разработок.

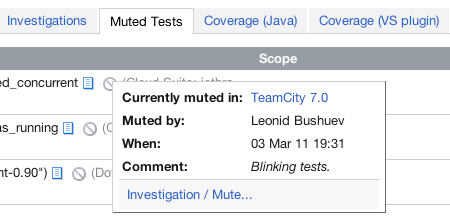
Средство командной строки - это экспериментальная функция и может быть установлена как отдельный плагин. В настоящий момент она поддерживает Perforce, Subversion, CVS и TFS.

А также:

* Возможности предварительного тестирования
* Расширение компонентов разработки
* Специализированное изменение страницы деталей/статуса
* Пошаговое обновление журнала разработок
* Архивирование проекта
* Резервное копирование и восстановление для лучшего сохранения TeamCity. Резервное копирование может быть запущено из пользовательского интерфейса или из командной строки; восстановление и миграция могут быть запущены только из командной строки. Резервное копирование командной строки не требует остановки работы TeamCity, если используется внешняя база данных.
* Проверка действий пользователя
* Включает поддержку .NET и Java. Основана на действующем движке для Java и NCover (версии 1,3) или PartCover для .NET.
* Удалённый запуск для Git из IntelliJ IDEA и Eclipse
* Поддержка Cucumber, Shoulda, Test-Spec в программе запуска Rake

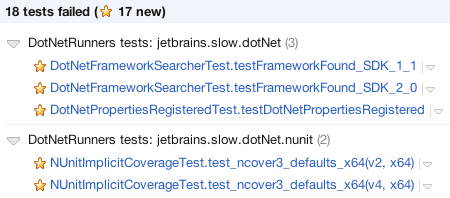
**Отключение неудачных тестов**

Программа отключает неудачные тесты, чтобы они не влияли на общее состояние сборки. Это позволяет значительно ускорить процесс сборки и избавить от лишнего «мусора». Данная функция особенно полезна, если некоторые тесты не удаются по известной причине, которую невозможно устранить в настоящий момент. Отключение тестов - потенциально опасная операция, доступная только для уровня прав «Администратор проекта» по умолчанию.



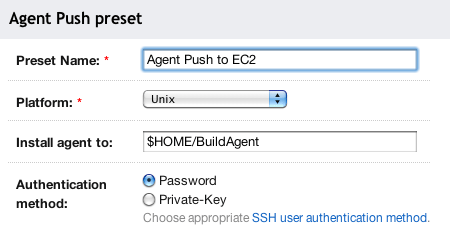
**Группировка тестов**

Группировка тестов по пакетам существенно ускоряет процесс выявления причин ошибок. Тесты могут быть сгруппированы в **TeamCity** на странице результатов сборки, текущей странице проблемы, страниц изменений и других.



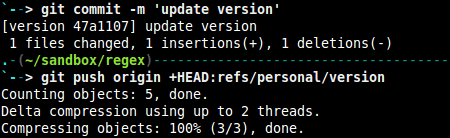
**Удаленная установка Агента**

**TeamCity** предоставляет варианты удаленной установки агентов сборки на машинах в сети или в облаке. Агент **TeamCity** может быть установлен на машинах Unix (по SSH) или Windows (с помощью Sysinternals PsExec). Можно настроить включение агента при запуске системы. Удаленная установка Агента интегрируется с облаком Amazon EC2 Cloud. **TeamCity** может автоматически устанавливаться на указанном агенте EC2.



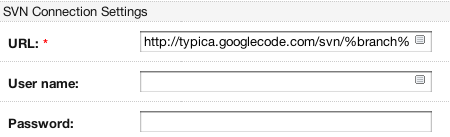
**Персональная сборка для Git и Mercurial**

**TeamCity** позволяет осуществлять персональную сборку на сервере с распределенными системами контроля версий. При использовании Git или Mercurial можно настроить конкретные конфигурации сборки для запуска в одной или нескольких ветвях VCS. Сборка осуществляется удаленно на сервере, при этом не влияет на глобальное состояние сборки и не появляется в истории у других пользователей.



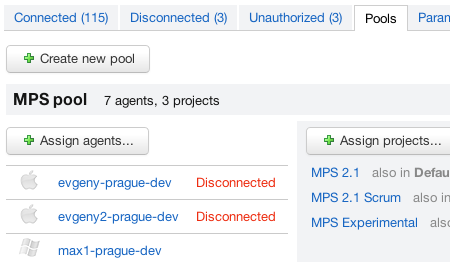
**Параметризованная система системы контроля версий**

VCS roots – наборы настроек VSC, используемые **TeamCity** для получения исходных файлов для сборки. VCS roots часто распределяется между различными проектами и конфигурациями сборки. **TeamCity** поддерживает параметризацию настроек VCS roots, что позволяет указывать необходимые параметры для конкретной конфигурации сборки. Это обеспечивает большую гибкость и возможность повторного использования параметров конфигурации.



**Пул агентов сборки**

Позволяет организовать агенты в пулы и назначать им сборку проекта. Пулы агентов можно быстро распространить между отделами или проектными группами, а также назначать их на основе приоритета выполнения конкретного проекта.

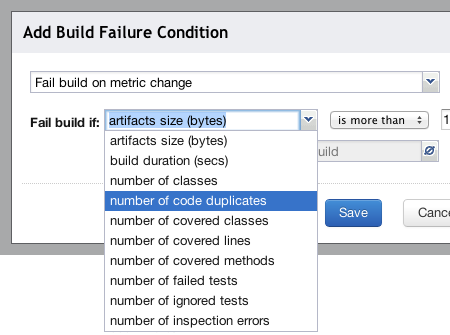


**Инкрементная сборка**

При инкрементной сборке компилируются только те файлы, которые претерпели изменения со времени последней инкрементной или полной сборки, что существенно сокращает время выполнения проекта. На данный момент эта функция доступна только для проектов Maven, Gradle и IntelliJ IDEA.

**Расширенные условия маркировки ошибок сборки**

Раньше возможность помечать неудавшиеся сборки была только в случае, если хотя бы один тест не удался. В новой версии были расширены условия признания сборки неудавшейся. Теперь можно помечать ошибки сборок по различным изменениям показателей или появлению особого текстового шаблона в логе сборки и т.д.



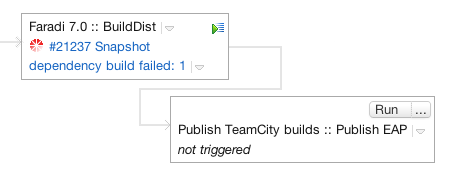
**Древовидный график для журнала изменений**

При работе с Git и Mercurial можно использовать интерактивный древовидный график на странице журнала изменений. Таким образом, есть возможность настроить удобный доступ ко всей информации об изменениях в проекте.



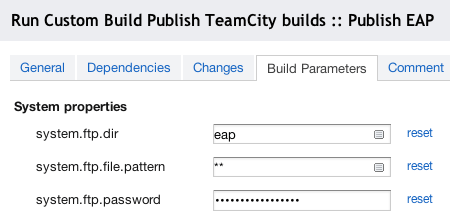
**Расширенная поддержка и визуализация функции Build chains**

Функция Build Chains позволяет разделять процесс сборки на этапы и выполнять обработку фрагментов можно получать подробную визуальную информацию о ходе выполнения всех процессов обработки.



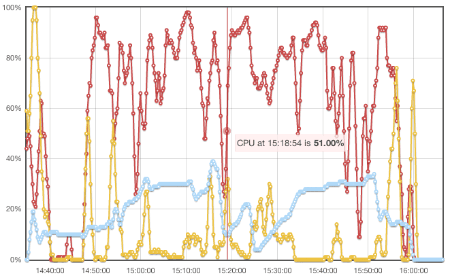
**Параметры сборки с пользовательским интерфейсом**

Возможность запускать пользовательские сборки всегда было одной из отличительных особенностей TeamCity. В новой версии можно указать определенный тип для любого пользовательского параметра в любой конфигурации сборки или проекта. На основе этого типа в диалоге Run Custom Build будет представлен определенный элемент управления для этого параметра, будь то текстовое поле, флажок, поле выбора или ввода пароля. При работе с паролем можно быть уверенным, что он будет надежно храниться и никогда не отобразится в журнале и других файлах TeamCity.



**Мониторинг производительности**

При расширении проекта количество модульных тестов возрастает до нескольких тысяч, что значительно снижает скорость выполнения сборок. Теперь доступна система мониторинга производительности, предоставляющая подробную информацию о задействовании аппаратных ресурсов на каждом этапе процесса сборки. На основе полученных данных можно планировать усовершенствование инфраструктуры.



**ReSharper на сервере**

Теперь механизм проверки ReSharper Code Analysis можно запускать на сервере. ReSharper Code Analysis позволяет обнаружить проблемы в коде C#, XAML, XML, ASP.NET и код.

**Интеграция с NuGet**

NuGet – удобный инструмент, позволяющий устанавливать, обновлять и удалять библиотеки (сборки), компоненты, инструменты. **TeamCity 7.0** тесно интегрируется с NuGet, поддерживая все его основные функции.

***Совместимость***

* *Maven 3 (по-прежнему поддерживается Maven 2)*
* *Subversion 1.7*

**Системные требования:**

**Server**

* Java (JRE) 1.5 и выше (Java 1.6 включена в поставку для Windows). TeamCity тестировался с Sun JDK. Both 32bit и 64bit JDK.
* J2EE Servlet (2.4 и выше) container, JSP 2.0 и выше container на Apache Jasper. TeamCity тестировался с Tomcat 6 (рекомендуется). Tomcat 6 также включен в поставку для Windows . TeamCity также работает с Jetty, Tomcat 5 и 7.
  + Но Tomcat версии 5.5.20 не совместим с TeamCity из-за ряда неисправностей и багов.
  + Рекомендуется использовать Tomcat 6.0.27 и выше, ранние версии могут вызвать проблемы при запуске TeamCity.
* Операционные системы:
  + - Linux
    - MacOS X
    - Windows XP
    - Windows Vista/Windows Vista 64
    - Windows 7/7x64
* Также запускается на:
* Solaris 10
* FreeBSD 7
* IBM z/OS

**Build Agent**

* Java (JRE) 1.5 и выше (Java 1.6 включена в поставку дляWindows; JDK (JRE не достаточно) потребуется для некоторых исполнителей).
* Операционные системы:
  + - Linux
    - MacOS X
    - Windows 2000/XP/XP x64/Vista/Vista x64
    - Windows 7/7x64
* Остановка процесса сборки доступна на:
  + - Windows 2000/XP/XP x64/Vista/Vista x64/7/7x64
    - Linux на x86, x64 процессорах PPC и PPC64
    - Mac OS X на процессорах Intel и PPC
    - Solaris 10 на x86, x64 процессорах

**Windows Tray Notifier**

Windows 2000/XP/Vista/Vista x64/7/7x64

**Браузеры**

* Microsoft Internet Explorer 6 и выше
* Mozilla Firefox 2, 3, 3.5, 3.6, 4 beta
* Opera 9.5, 10.0
* Safari 3 и выше на Mac/Windows
* Google Chrome на Mac/Windows

**Build Runners**

Поддерживаемые Java исполнители:

* Ant 1.6-1.8 (в комплекте Ant 1.8.0)
* Maven2 (2.0.x, 2.x) (в комплекте Maven 2.2.1)
* IntelliJ IDEA исполнитель
* Gradle (необходим Gradle 0.9-rc-1 и выше)
* Java 1.6

Поддерживаемые .Net исполнители:

* MSBuild (требуется .Net Framework или Mono, установленный вместе с build agent)
* NAnt 0.85 - 0.90 alpha 1 (требуется .Net Framework или Mono, установленный вместе с build agent)
* Microsoft Visual Studio Solutions (2003, 2005, 2008, 2010) (требуется MS Visual Studio, установленная вместе с build agent)
* FxCop (требуется FxCop , установленный вместе с build agent)
* Duplicates Finder для проектов Visual Studio 2003, 2005, 2008 и 2010. Поддерживаются языки C# до версии 4.0 и Visual Basic .Net версий 8.0 - 10.0 (требуется .Net Framework 2.0 и выше , установленный вместе с build agent)

Другие исполнители:

* Rake
* Исполнитель командной строки

**Testing Frameworks**

* JUnit 3.8.1 и выше, 4.x
* NUnit 2.2.10, 2.4.x, 2.5.x (dedicated build runner)
* TestNG 5.3 и выше
* MSTest (dedicated build runner; требуется соответствующая версия Microsoft Visual Studio edition, установленная вместе с build agent)

**Version Control Systems**

Checkout на сервере

* Borland StarTeam 6 и выше (клиент StarTeam должен быть установлен на сервере TeamCity)
* CVS
* IBM Rational ClearCase, Base и UCM (клиент ClearCase должен быть установлен на сервере TeamCity)
* Microsoft Visual SourceSafe 6 и 2005 (клиент SourceSafe должен быть установлен на сервере TeamCity, только Windows)
* Perforce (клиент Perforce должен быть установлен на сервере TeamCity)
* Subversion
* Team Foundation Server 2005, 2008 и 2010 (Team Explorer должен быть установлен на сервере TeamCity, доступно только на Windows)
* Git
* Mercurial (клиент Mercurial "hg" должен быть установлен на сервере)
* SourceGear Vault (клиентские библиотеки Command line должны быть установлены на сервере)

Checkout на агенте

* CVS
* Perforce
* Clearcase
* Subversion
* Team Foundation Server 2005, 2008 и 2010 (Team Explorer должен быть установлен с build agent)
* Mercurial (клиент Mercurial "hg" должен быть установлен с build agent)
* Git

Маркировка исходников

* Borland StarTeam
* ClearCase
* CVS
* Perforce
* Subversion
* Mercurial
* Git
* Team Foundation Server

Системы отслеживания ошибок

* JetBrains YouTrack 1.0 и выше.
* Atlassian Jira 3.10 и выше.
* Bugzilla 3.0 и выше.

**Интеграция с IDE**

* Eclipse 3.2-3.6, с JDK 1.5 и выше
* Плагин для платформы IntelliJ: совместим с IntelliJ IDEA 8.x, IntelliJ IDEA 9.0.x, 10.0 (версии Ultimate и Community); JetBrains RubyMine 2.0, 3.0, Jetbrains PyCharm 1.0 и выше, Jetbrains PhpStorm 1.0 и выше, WebStorm 1.0 и выше
* Microsoft Visual Studio 2005, 2008 и 2010

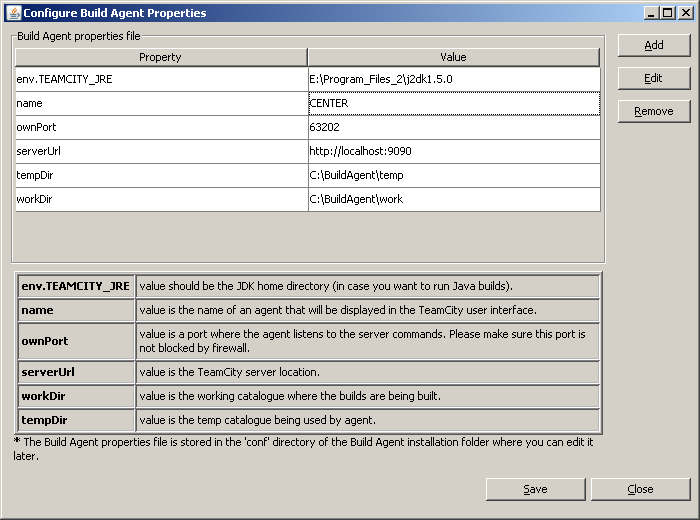
**Внешние базы данных**

* HSQLDB
* MySQL 5.0.33 и выше
* Microsoft SQL Server 2005, 2008
* PostgreSQL 8 и выше
* Oracle 10g и выше

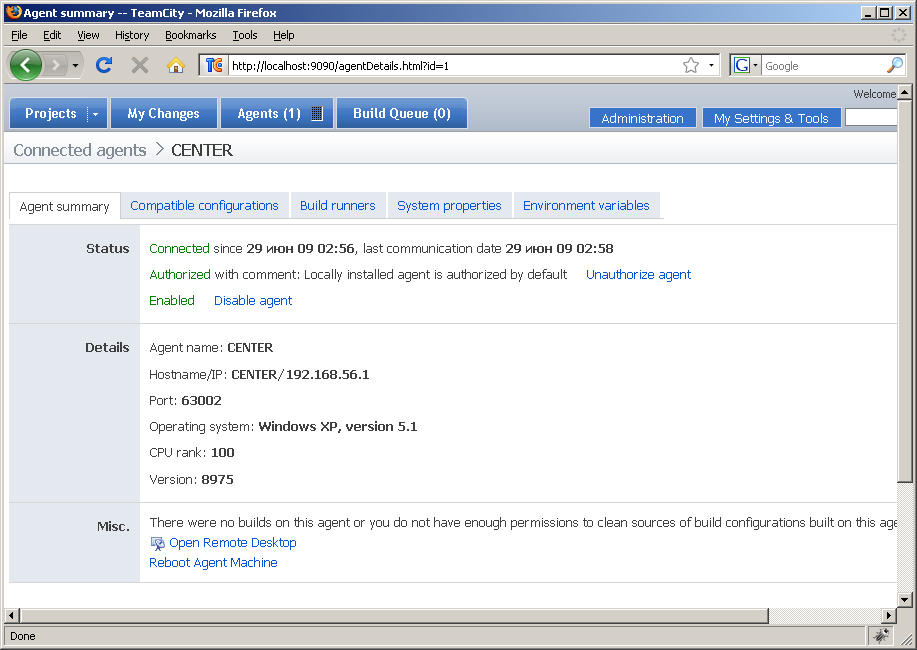
1. Настройка Teamcity.

Teamcity условно состоит из двух частей: веб-сервер и билд-агент. Так, после установки на компьютер teamcity, в списке зарегистрированных служб windows появятся два новых пункта: “TeamCity Web Server” и “TeamCity Build Agent Service”. Первая их них представляет собой веб-сервер (tomcat 6), на котором размещен веб-интерфейс для teamcity. С помощью этого интерфейса можно создавать конфигурации билдов, запускать и останавливать их, наблюдать за прогрессом операций и отслеживать составляющие билд правки из cvs/svn-репозитория и многое-многое другое. Но, и это важно, сам процесс сборки проекта не является зоной ответственности для “TeamCity Web Server”: за сборку проекта отвечает “TeamCity Build Agent Service”. Конфигурация “один веб-сервер и один билд-агент” является самой простой. Гораздо чаще применяется конфигурация “один веб-сервер и отдельный билд-агент для каждой билд-конфигурации”. Т.е. teamcity фактически может быть распределена на нескольких компьютерах в сети. Один из компьютеров, на котором размещен “TeamCity Web Server”, играет роль дирижера, управляющего работой остальных машин с билд-агентами. Каждый раз, когда кто-нибудь инициирует билд, то “TeamCity Web Server” связывается с билд-агентами и узнает, кто из них может выполнить билд (обладает техническими возможностями). А затем билд ставится в очередь выполнения для того агента, который сейчас свободен и не занят другой работой.

Первым шагом после установки teamcity, нужно зайти в панель управления службами windows и запустить две службы, составляющих teamcity: web server и build agent. Теперь можно открыть в браузере адрес http://localhost:9090 (номер порта указывается при установке teamcity) и на первой страничке teamcity сразу же попросит создать учетную запись администратора. После того как учетная запись администратора была создана, и был выполнен вход внутрь teamcity, появляется интерфейс с шестью закладками: “Projects”, “My Changes”, “Agents”, “Build Queue”, “Administration” и “My Settings & Tools”. Основное внимание на вкладку “Agents”. На ней находится перечень обнаруженных teamcity компьютеров с установленными билд-агентами. Изначально, в перечне будет только один агент, установленный вместе с teamcity (имя агента по-умолчанию совпадает с именем компьютера). На вкладке “Agents” есть ссылки на загрузку инсталляционных файлов билд-агентов. Это может быть либо классический MS Windows Installer, либо инсталлятор в виде исполняемого файла Java Web Start (при работе с linux). В любом случае, после того как инсталлятор агента загружен и выполнен, нужно указать каталог, куда будет установлен агент. В этом же каталоге будет выполняться и, собственно, сборка проекта, так что нужно достаточное количество места. Затем появится диалоговое окно, в котором нужно указать параметры коммуникации агента с веб-сервером teamcity.

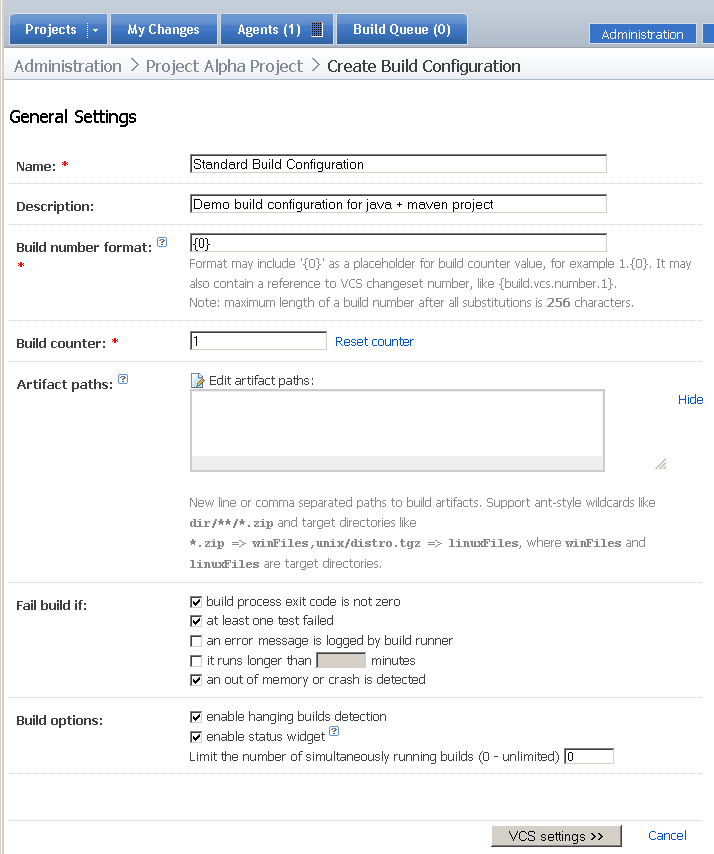
[](http://study-and-dev.com/blog/wp-content/uploads/mediawiki/images/upravlenie_sborkoy_proektov_vmeste_s_teamcity_chast_1/Teamcity_1_2.png)

Из основных настроек нужно задать номер порта, на котором будет размещен билд-агент (ownPort) и адрес веб-сервера teamcity (serverUrl). После завершения установки агента и его запуска, в интерфейсе вебс-вервера teamcity должны появиться все настроенные агенты. Все зарегистрированные и работоспособные сервера находится в таблице “Connected agents”.

[](http://study-and-dev.com/blog/wp-content/uploads/mediawiki/images/upravlenie_sborkoy_proektov_vmeste_s_teamcity_chast_1/Teamcity_1_3.png)

Если же какой-то агент в текущий момент времени не может общаться с teamcity веб-сервером, то он попадает внутрь списка “Disconnected agents”. Каждый из билд-серверов может быть отключен на какое-то время и включен обратно, также можно отзывать с билд-серверов статус “авторизован”. На вкладке с билд-агентами можно просмотреть статистику их работы, если открыть вкладки “Matrix” и “Statistics”.

Для создания нового teamcity-проекта необходимо зайти в панель управления teamcity от имени администратора и затем перейти на закладку “Administration”. Там расположен список всех созданных проектов и привязанные к ним билд-конфигурации. Для создания нового проекта используется функция “Create project”. В появившемся диалоговом окне создания нового проекта нужно указать имя проекта и произвольный комментарий описания. Фактически, проект – это способ группировки нескольких билд-конфигураций в единую административную единицу. Например, разрабатывая проект (назовем его “MultiProject”) для нескольких платформ, например, windows и linux, создается две билд-конфигурации с различными настройками, и выполняющиеся под различными билд-агентами. Т.е. билд-агент на windows машине выполняет сборку проекта для windows, а билд-агент на lunux-машине выполняет, соответственно, сборку того же проекта, но для linux-окружения. Для присоединения к проекту билд-конфигурации используется кнопка “Create build configuration”. Для настройки билд-конфигурации необходимо пройти через ряд последовательных шагов мастера создания билд-конфигурации. Первый экран мастера настройки называется “General Settings”.

[](http://study-and-dev.com/blog/wp-content/uploads/mediawiki/images/upravlenie_sborkoy_proektov_vmeste_s_teamcity_chast_2/Teamcity_2_1.png)

Единственные очевидные параметры из указываемых на этом экране это “Name” и “Description”, т.е. название билд-конфигурации и комментарий к ней. Параметры “Build number format” и “Build counter” должны рассматриваться совместно: первый из них – это шаблон, согласно которому будут нумероваться билды и, в самом простом случае, номер билда – это просто число. Комбинация символов “{0}” означает, что вместо них teamcity подставит в номер билда число, увеличивающееся при каждом очередном билде на единицу. Текущее значение этого числа как раз и хранится в поле “Build counter”. Графа “Artifact paths”. Каждая сборка по своему завершению может генерировать несколько выходных файлов (в терминологии teamcity артефактов). Самый очевидный пример артефакта – это сам финальный файл билда, т.е. приложение, которое затем поступит для тестирования. Другими примерами артефактов могут быть отчеты о ходе компиляции, отчеты о результатах автоматического тестирования встроенного в билд-цикл. Фактически, любой файл или каталог, формируемый в ходе билда, может стать артефактом. После того как билд будет завершен и всем заинтересованным участникам будет разослано сообщение об этом, то пользователь, зашедший на страничку с уведомлением о результатах билда, увидит также перечень ссылок позволяющих скачать созданные артефакты. Следующая графа “Fail build if” служит для того, чтобы задать перечень критериев, позволяющих определить то, что билд был провален. Традиционно, все билд-утилиты извещают о результате своего завершения через код возврата “build process exit code is not zero” . Также можно пометить билд как “проваленный”, если был провален хотя бы один из автоматических тестов “at least one test failed”, или если билд выполняется слишком долго (завис) “it runs longer than X minutes”. Опция “enable hanging builds detection” – включить режим обнаружения “зависших билдов”.

Следующая страница “Version Control Settings” служит для управления правилами интеграции teamcity и системы управления версиями документов. Первым шагом нужно настроить VCS Root, т.е. зарегистрировать cvs/svn репозиторий, указав для него путь и учетную запись для подключения. Для этого нужно нажать на кнопку “Create and attach new VCS root”. На странице “New VCS Root” нужно указать название VCS Root и выбрать из падающего списка тип используемой системы хранения версий документов. После выбора “Subversion”, на экране появится ряд опций настройки подключения к svn-репозиторию. В графе “URL” указывается, не полный путь к, непосредственно, каталогу с исходными кодами проекта (например, они расположены по адресу http://localhost:8080/svn/demo1/trunk/tcitydemo1/), а к корню репозитория, выделенного для хранения проекта http://localhost:8080/svn/demo1/. Дело в том, что в терминологии teamcity “VCS Root” не является непосредственным адресом, по которому расположен проект или его HEAD-версия. Скорее, VCS Root является “точкой доступа”, т.е. адрес сервера, учетные записи для доступа и сведения о структуре репозитория. Далее, в графах “User name” и “password” указывается имя и пароль пользователя имеющего право работать с svn-репозиторием. После создания VCS Root, возвращаемся на страницу настройки VCS, где можно указать еще несколько настроек. Во-первых, “checkout rules”, т.е. правила по которым из CVS Root будут извлечены каталоги с исходными кодами проекта. Для этого используется кнопку “edit checkout rules”. Открывается диалоговое окно с правилами “отображения” структуры svn-репозитория на каталоги build-агента.

+:trunk/tcitydemo1=>.

Указанное выше правило говорит, что нужно извлечь все файлы и подкаталоги, расположенные внутри “trunk/tcitydemo1” (перед префиксом добавляется адрес VCS Root) и поместить их в корень каталога копии проекта. Можно записать несколько правил импорта файлов из репозитория, можно комбинировать эти правила, например, исключая из импорта файлы по определенной маске и т.д. Опция настройки VCS – “VCS checkout mode” – имеет следующие значения. Во-первых, “Automatically on Server” предполагает, что работа с svn-репозиторием будет выполнена на teamcity сервере, а затем файлы проекта будут скопированы на машины с билд-агентами. Как побочный эффект этой стратегии на машины с билд-агентами не будут скопированы служебные файлы svn (скрытый каталог “.svn”). Вторая возможная стратегия “Automatically on Agent” предполагает, что файлы из svn-репозитория будут загружены не сервером teamcity, а билд-агентом (в этом случае, служебный каталог “.svn” будет доступен). Третья стратегия “Do not checkout files automatically” предполагает, что файлы из репозитория должны быть доступны заранее до начала, собственно, процесса сборки проекта. “Clean all files before build” служит для предварительной очистки каталогов с исходными кодами проекта, перед тем как эти файлы будут заново извлечены из svn-репозитория. Одна из самых полезных функций связанных с интеграцией с svn является возможность маркировать метками “удачные” билды. Т.е. предположив, что какой-то билд был удачен, мы можем сделать в svn-репозитории копию файлов образующих этот билд. В последствии всегда можно вернуться именно к этому состоянию репозитория и проекта. Или можно сравнить какие файлы были изменены между двумя произвольными метками. Билды, точнее образующие их файлы проекта, могут маркироваться либо автоматически, либо вручную. Для автоматического режима нужно включить опцию “VCS labeling mode”, установив ее в “Successful only”, а в графе “Labeling pattern” указывается правило назначения имени метки, например, “label\_%system.build.number%”. Здесь вместо “%system.build.number%” будет подставляться номер билда.

Третий шаг настройки конфигурации билда – выбор “Build Runner”. Именно здесь указывается какая утилита выполняет фактическую сборку проекта. В графе “Goals:” указывается список целей, которые нужно выполнить maven-у. Графа “Maven home path” позволяет указать путь к каталогу, где на машине с билд-агентом установлен maven. Большой необходимости в том, чтобы заполнять это поле нет, т.к. в инсталляционном пакете teamcity уже идет интегрированный maven 2.0.9 (для teamcity 4.5.1). “Additional Maven command line parameters” позволяют указать дополнительные параметры командной строки для maven. Если необходимо использовать нестандартный файл settings.xml, то путь к нему указывается в графе “User settings path”. В том случае, если необходимо выполнять сборку проекта под версией java отличной от установленной по умолчанию на машине с билд-агентом, то нужно указать путь к ней в графе “JDK home path”. Поле “JVM command line parameters” позволяет настроить специфические параметры java-машины используемой для запуска maven-сценария (например, увеличить лимит оперативной памяти резервируемой для билд-скрипта). Teamcity достаточно умен для того, чтобы распознать результат выполнения встроенных в жизненный цикл проекта автоматических тестов и, например, пометить билд как неудачный, если тесты были провалены. Однако, можно подсказать teamcity, где найти результаты запуска тестов и что с ними делать с помощью опций “XML Report Processing”.

Остальные настройки билд-конфигурации указывать хоть и не обязательно, но полезно. Прежде всего, можно настроить правила автоматического запуска билдов. По-умолчанию, для запуска билда пользователь, наделенный соответствующими правами должен зайти в панель управления teamcity, выбрать в списке доступных билд-конфигураций нужную и нажать кнопку “Run”. Помимо этого, удобным является настройка запуска билда по расписанию, например, каждый день в 23:00 или каждую пятницу в 23:00 и т.д. Количество подобных настроек времени срабатывания никак не ограничено. Еще один способ настройки автоматического запуска билдов связан с настройкой зависимости одной билд-конфигурации от другой. К пример, после того как билд-конфигурация “A” была запущена и отработала, то автоматически стартует конфигурация “B”. Еще одна полезная функция связана с отслеживанием teamcity изменений в svn-репозитории и автоматическом запуске билда. Настраивается все это на закладке “VCS Triggers”. Там можно указать несколько шаблонов имен файлов внутри svn-репозитория, при изменении которых teamcity начинает “предстартовый” отсчет перед запуском билда. Для того чтобы не запускать билд, когда не все еще файлы были сохранены в репозиторий, teamcity ждет чтобы в течении “Quiet period mode” времени не было никаких правок в репозитории и только после этого билд стартует. Следующая опция настройки билд-конфигурации размещена на вкладке “Dependencies” и служит для настройки списка предварительных условий, перед тем как билд будет запущен. Условно говоря, можно настроить три билд-конфигурации “A”, “B” и “C” так, чтобы после того как конфигурация “A” была “собрана”, то из нее будут извлечены артефакты (файлы или каталоги), которые затем будут скопированы в рабочий каталог для конфигурации “C”. Если в тот момент, когда мы запустили конфигурацию “C”, зависящую от конфигураций и артефактов “A” и “B”, еще не были готовы результаты их работы. То teamcity либо подождет завершения поставленных в очередь заданий “A” и “B” либо инициирует их запуск.

Закладка “Properties and environment variables” служит для того, чтобы передать внутрь сценария сборки проекта какие-то внешние данные. К примеру, можно записать внутри maven-сценария зависимость от junit-библиотеки указав ее версию так:

1. **<dependency>**
2. **<groupId>**junit**</groupId>**
3. **<artifactId>**junit**</artifactId>**
4. **<version>**${JUNIT.VERSION}**</version>**
5. **<scope>test</scope>**
6. **</dependency>**

Т.е. должна существовать какая-то переменная “${JUNIT.VERSION}” хранящая значение версии библиотеки junit.

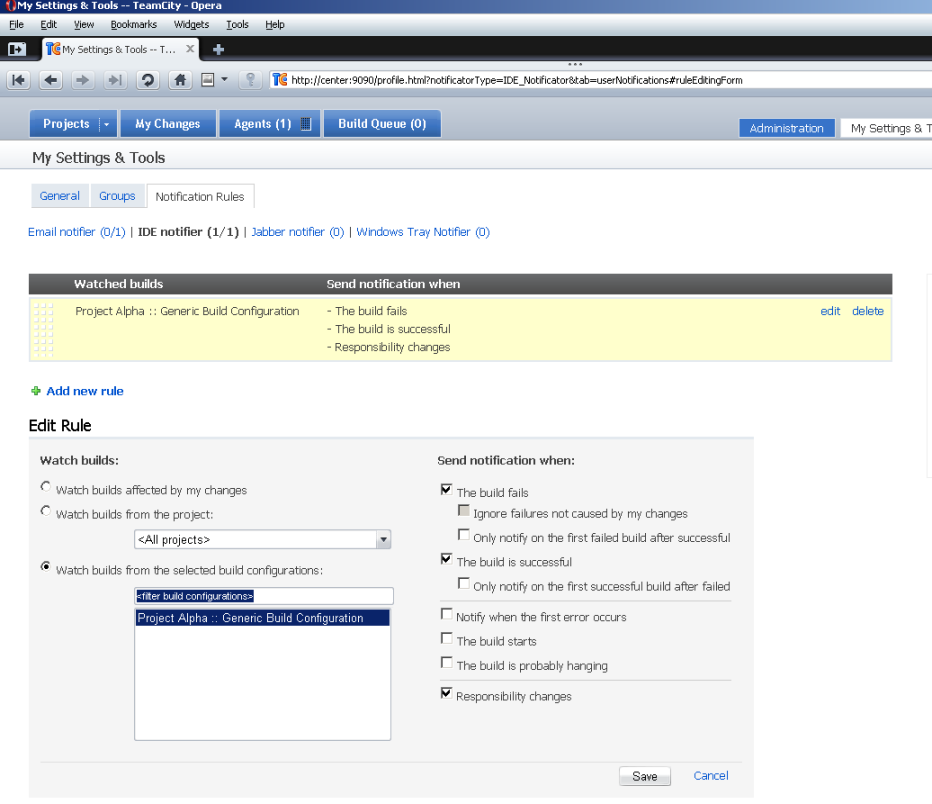
После того как конфигурация была полностью определена, можно перейти на главную страницу панели управления teamcity и для созданной только что конфигурации выполнить команду запуска “Run”.

Teamcity отображает прогресс выполнения билда, одновременно с этим примерно оценивая уже прошедшее и еще требуемое для завершения билда время (на основании статистического анализа времени выполнения предыдущих билдов). Также показаны для билда сведения о том, какие пользователи внесли изменения в файлы проекта и даже информация о версиях самих файлов. Затем после завершения билда можно просмотреть на вкладке “Changes” как общую информацию об измененных для этого билда файлах, так и в удобном diff-редакторе отличия между файлами.

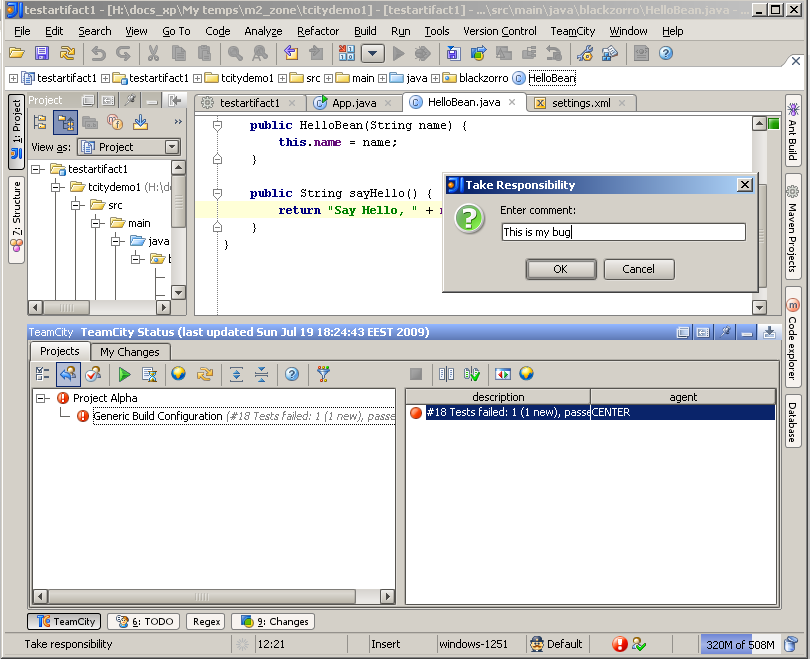
После того как свободный билд-агент закончит сборку maven-проекта, будет создано несколько отчетов. Во-первых, отчет о результатах запуска автоматических тестов, затем идет отчет о списке изменений в репозитории cvs/svn, которые были включены в текущую сборку. И, в конце концов, полный журнал всех сообщений, которые были сгенерированы билд-сценарием. Что касается сроков хранения артефактов, равно как и хранения истории запуска билдов, то это можно настроить в панели управления teamcity на странице “Administration -> Clean-up process settings”.

**Интеграции teamcity со средой разработки intellij idea.**

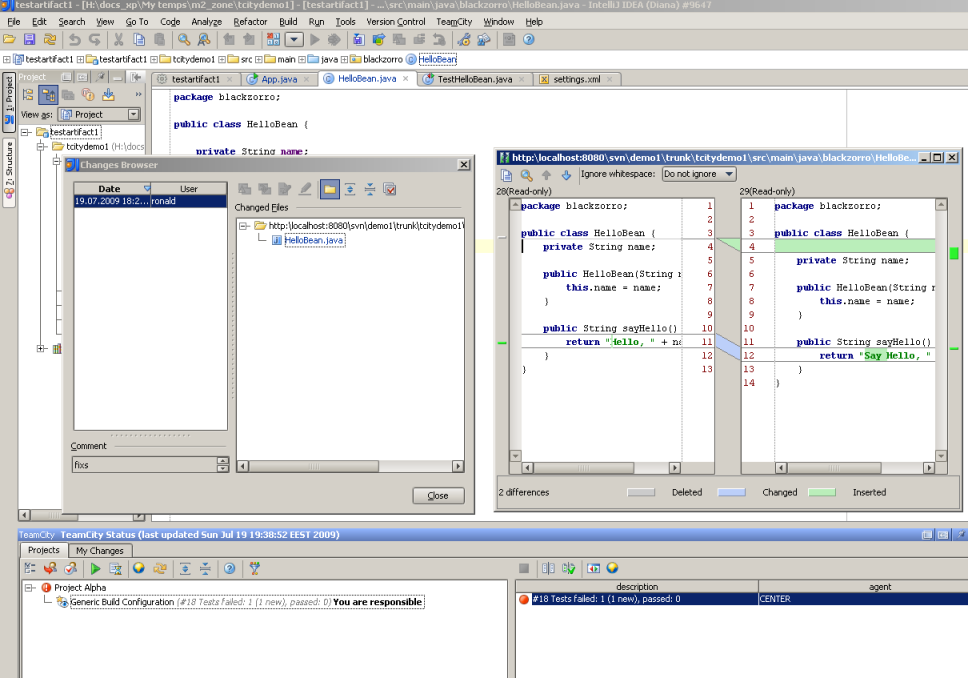
Для того, чтобы установить к себе на компьютер плагин teamcity, необходимо зайти в панели управления teamcity на страницу “My Settings & Tools”, где расположены ряд ссылок для загрузки плагинов. Скачать файл “TeamCity-IDEAplugin.zip ” плагина для idea, затем распаковать содержимое архива в каталог idea/plugins, и после перезапуска среды в idea получит в свое распоряжение новое меню “teamcity”. Первым шагом настройки интеграции idea и teamcity является регистрация пользователя. Для этого нужно в меню “Teamcity -> Login” и в появившемся диалоговом окне указать адрес подключения к teamcity, свое имя и пароль. Teamcity может посылать сообщения о таких событиях как запуск билда и результате его выполнения, например, по email или jabber клиенту, но гораздо удобнее если извещения показываются в среде разработки т.к. появляется дополнительный функционал быстрого перехода к списку изменений произошедших в проекте с момента прошлого билда. Итак, выбрав меню “teamcity -> Edit Notification Rules” затем в открывшемся окне браузера вводим свое имя и пароль, чтобы попасть внутрь teamcity и, наконец, открывается страница редактирования правил извещений.

[](http://study-and-dev.com/blog/wp-content/uploads/mediawiki/images/upravlenie_sborkoy_proektov_vmeste_s_teamcity_chast_3/Teamcity_3_2.png)

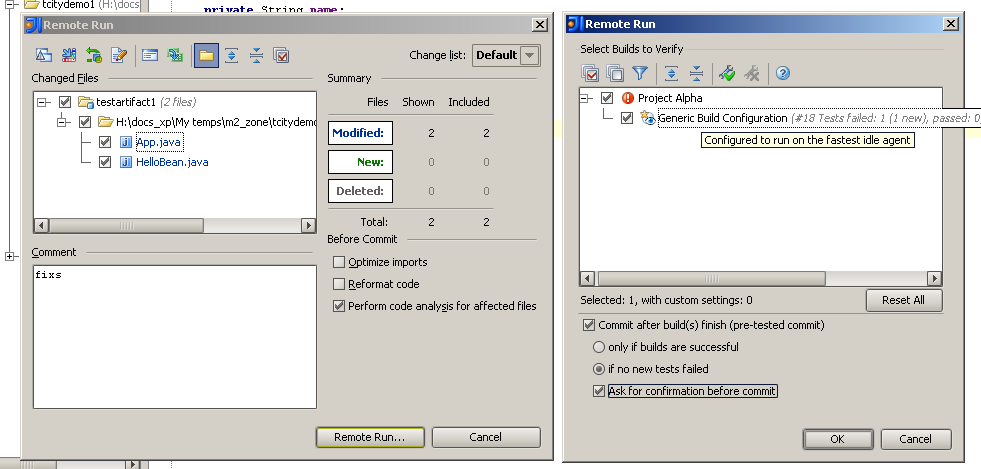
Закладка “IDE notifier” позволяет выбрать то, когда teamcity пошлет нам извещение. Выбраны события “The build is successful”, “Responsibility changes” и “The build fails”. Теперь после того как билд будет завершен, появится всплывающее окошко с сообщением об этом. Более того, teamcity плагин предоставляет доступ к панели наблюдения за статусом билдов.

[](http://study-and-dev.com/blog/wp-content/uploads/mediawiki/images/upravlenie_sborkoy_proektov_vmeste_s_teamcity_chast_3/Teamcity_3_3.png)

На рисунке показан момент, когда сборка проекта завершилась неудачей, и разработчик решил взять на себя ответственность за этот сбой. Т.е. он сообщает teamcity о том, что именно его изменения являлись причиной сбоя. После того как ошибка в коде будет исправлена и сохранены изменения в cvs|svn, то можно отметить следующий билд как содержащий исправления для предыдущего. Как приятный бонус после установки плагина интеграции teamcity и idea появляется функция быстрого перехода из окна браузера, например, со страницы с перечнем измененных файлов, в окно java редактора. Также панель teamcity содержит функцию просмотра списка последних файлов и позволяет просмотреть различия между ними с помощью удобного diff-редактора.

[](http://study-and-dev.com/blog/wp-content/uploads/mediawiki/images/upravlenie_sborkoy_proektov_vmeste_s_teamcity_chast_3/Teamcity_3_4.png)

Крайне полезна в практике и функция “Remote Run”. Так если внутри редактора idea изменить какой-то файл и выбрать в меню пункт “teamcity -> remote run” то idea отправит файлы с изменениями кода не в cvs/svn репозиторий, а непосредственно на машину с билд агентом.

[](http://study-and-dev.com/blog/wp-content/uploads/mediawiki/images/upravlenie_sborkoy_proektov_vmeste_s_teamcity_chast_3/Teamcity_3_5.png)

На рисунке показан пример диалогового окна, с настраиваемыми параметрами Remote Run, где указано, что если запуск билда с правками будет успешен и ни один из тестов не будет провален, то изменения нужно автоматически сохранить и в cvs/svn репозиторий.

1. Список используемых источников
2. <http://www.jetbrains.com/teamcity/>
3. Ильин А. TeamCity: распределенное управление сборкой проектов / Веб-Аналитик.ИНФО. - 2010 - №11.
4. Управление сборкой проектов вместе с teamcity / Компьютерная газета. – 2009.