



PLARIUM

AUTOMATION BUILDS WITH CAKE

ЧТО, ЗАЧЕМ И КАКИМ ОБРАЗОМ

Артем Дурнев
Unity Software Architect



UD

TECH
MEET
UPS

V

СОДЕРЖАНИЕ

- 1 Что такое **Cake**. Описание и ключевые преимущества
- 2 Основы работы с **Cake**
- 3 Разбор примеров использования **Cake** в наших проектах

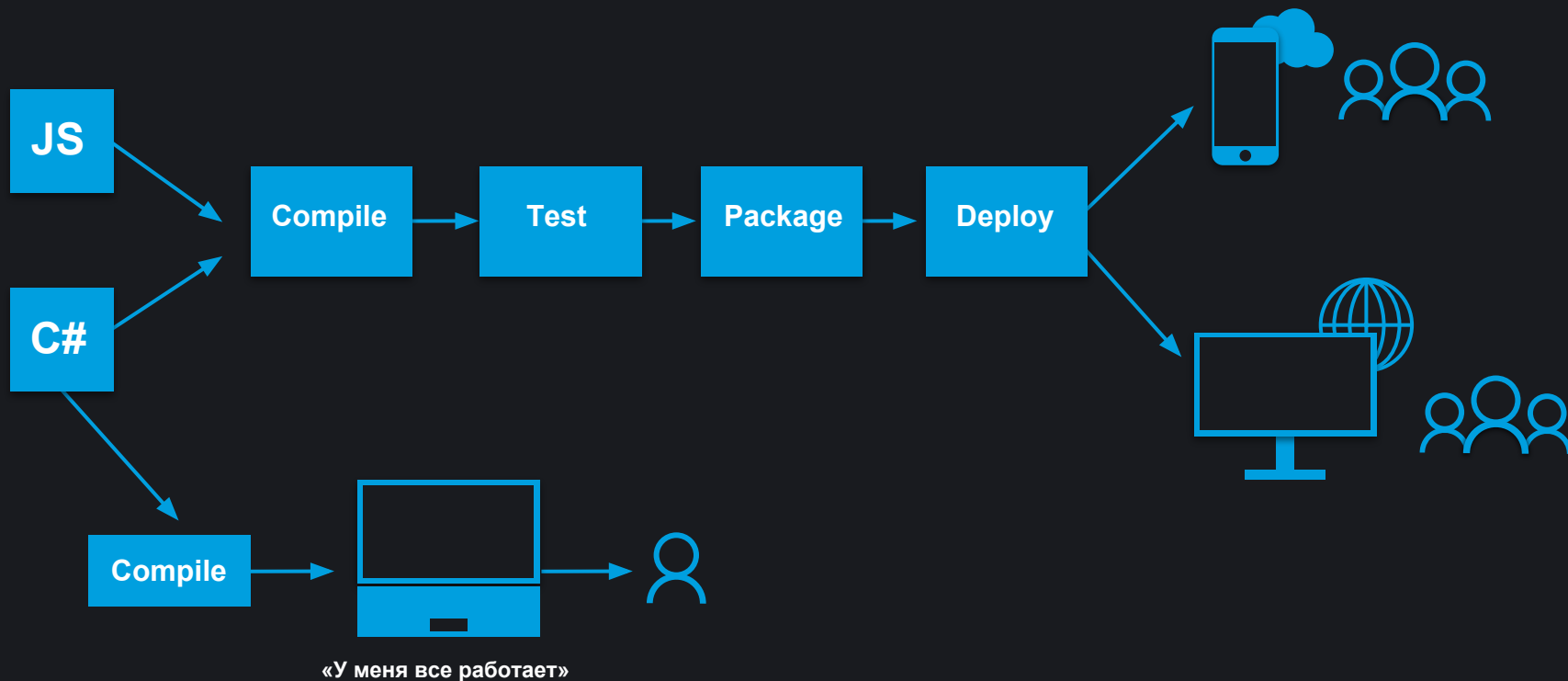


CAKE. ЧТО ЭТО?

ОПИСАНИЕ И КЛЮЧЕВЫЕ ПРЕИМУЩЕСТВА



ПРОБЛЕМА



ПРОБЛЕМА

- Несколько языков разработки
- Множество стадий билд процесса
- Различное поведение в зависимости от окружения

ПРОБЛЕМА – COMPILE

- Чекаут из разных репозиториев
- Восстановление пакетов npm / nuGet
- Разрешение зависимостей
- Версионирование
- Обфускация кода
- Ветвления логики компиляции



ПРОБЛЕМА – TEST

- Unit тестирование
- Определение покрытия кода тестами
- Статический анализ кода
- UI тесты

ПРОБЛЕМА – PACKAGE

- Подпись кода
- Создание релиз ноутов
- Генерация документации



ПРОБЛЕМА – DEPLOY

- Zip
- Миграция баз данных
- Работа с FTP
- Работа с пулами приложений
- Нотификации про выполнение операции

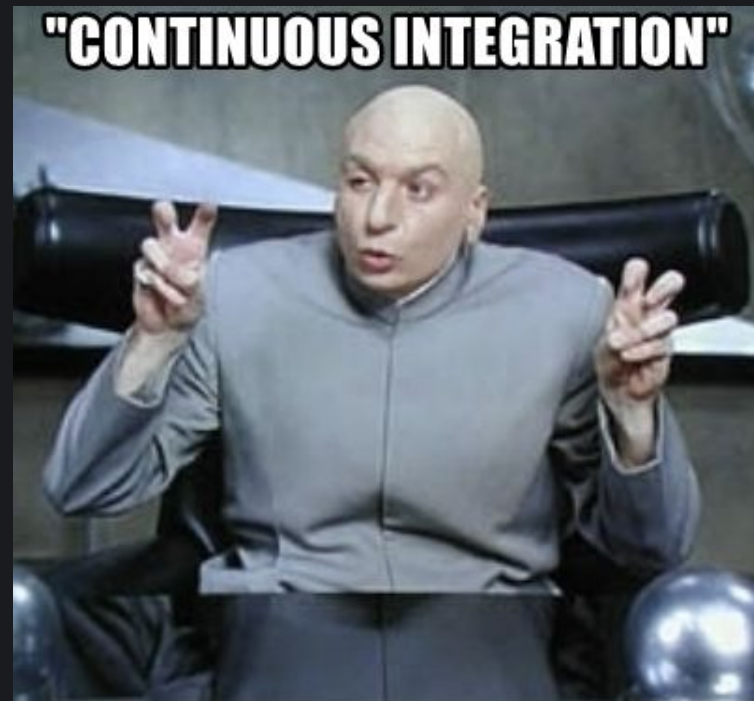
АЛЬТЕРНАТИВЫ

- Попросить вылить того, кто уже выливал
- CI сервер
- Написать свой скрипт



АЛЬТЕРНАТИВЫ – РУКАМИ

- Каждый раз по новому
- «У меня все работает»
- Нет никакой периодичности
- Занимает много времени
- Отпуска и больничные
- Через пол года уже никто ничего не помнит



АЛЬТЕРНАТИВЫ – CI СЕРВЕР

- Сложно дебажить процесс билда
- Конфигурация находится не под version-контролем
- Нет связи между решением и определением процесса билда
- Для некоторых CI систем новый синтаксис



АЛЬТЕРНАТИВЫ – СВОЙ СКРИПТ

- PowerShell – мощный язык, но им нужно владеть
- Проблемы с мультиплатформенность (ее нет)
- Огромное количество подводных камней

КЛЮЧЕВЫЕ ПРЕИМУЩЕСТВА

Cake (C# Make) is a cross-platform build automation system with a C# DSL for tasks such as compiling code, copying files and folders, running unit tests, compressing files and building NuGet packages.



КЛЮЧЕВЫЕ ПРЕИМУЩЕСТВА

- Полный контроль над процессом билда
- Мультиплатформенность – поддерживается запуск на Windows, Linux, macOS
- Для написания кода используется C# с всеми его преимуществами
- Легкость расширения функциональности через nuGet
- Open Source. Разработка ведется на GitHub
- Есть каналы для общения с мейнтейнерами в Gitter
- Огромная экосистема. 112 расширений для любых потребностей, среди которых MSBuild, Git, SemVer2.0.0, etc



CAKE. ОСНОВЫ РАБОТЫ



СТРУКТУРА ФАЙЛОВ ДЛЯ CAKE

- *build.ps1/build.sh*
 - Bootstrapper скрипты, которые обеспечивают установку зависимостей для Cake и правильный запуск
 - Написаны на PowerShell для Win и Bash для Linux/Mac
 - Обеспечивают валидацию параметров и запуск нужного таска на выполнение
 - Может быть изменен под нужды проекта
- *build.cake*
 - Основной скрипт, написанный на C# с некоторыми особенностями
 - Содержит основную логику
 - Имя может отличаться от стандартного
- *tools/packages.config* (не обязательно)
 - Говорит bootstrapper скрипту, какие nuGet пакеты и какой версии устанавливать перед запуском

СТРУКТУРА ФАЙЛОВ ДЛЯ CAKE

The image shows a screenshot of the Visual Studio Code interface. On the left, the Explorer sidebar is open, showing the file structure of a project. The 'ANALYSIS' section is expanded, and the 'packages.config' file is selected. The file structure includes:

- EXPLORER
 - OPEN EDITORS
 - packages.config tools
 - ANALYSIS
 - .vscode
 - artifacts
 - packages
 - src
 - tools
 - Cake
 - nuget.exe
 - packages.config**
 - .gitattributes
 - .gitignore
 - build.cake
 - build.ps1
 - build.sh

On the right, the Editor window shows the content of the 'packages.config' file, which is an XML configuration file:

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <packages>
3   <package id="Cake" version="0.27.1" />
4 </packages>
5
```

IDE

- **Visual Studio**

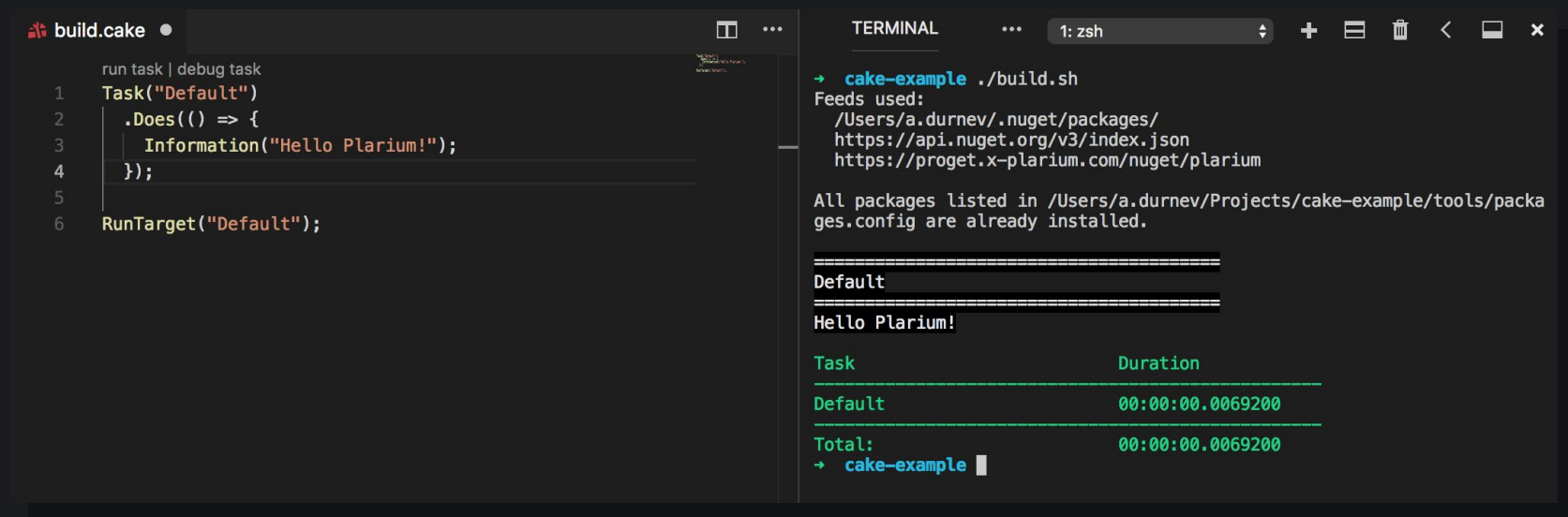
- Нужно установить расширение
- Поддержка запуска задач на выполнение, с выводом во внутреннюю консоль
- Поддержка Debug
- Нет intellisense, но есть подсветка синтаксиса

- **VS Code**

- Нужно установить расширение
- Поддержка запуска задач на выполнение, с выводом во внутреннюю консоль
- Поддержка Debug
- Подсветка синтаксиса и intellisense
- Дополнительные команды и сниппеты
- ИМХО работает не всегда идеально, особенно на Mac

TASKS

- Основной юнит выполнения **Cake**
- Могут иметь зависимости, критерии и специфическую обработку ошибок



```
build.cake •
```

```
1 run task | debug task
2 Task("Default")
3 | .Does(() => {
4 |     Information("Hello Plarium!");
5 | });
6 RunTarget("Default");
```

```
TERMINAL 1: zsh
```

```
→ cake-example ./build.sh
Feeds used:
/Users/a.durnev/.nuget/packages/
https://api.nuget.org/v3/index.json
https://proget.x-plarium.com/nuget/plarium

All packages listed in /Users/a.durnev/Projects/cake-example/tools/packa
ges.config are already installed.

=====
Default
=====
Hello Plarium!

=====
Task                                     Duration
-----
Default                                00:00:00.0069200
Total:                                  00:00:00.0069200
→ cake-example
```

TASKS

- Указывает какой task будет запускаться
- Всегда должен быть в конце
- У задач нет параметров

 build.cake x

```
run task | debug task
1 Task("Default")
2   .Does(() => {
3     |   Information("Hello Plarium!");
4   });
5
6 RunTarget("Default");|
```

ЗАВИСИМОСТИ

- Таски могут быть зависимы друг от друга
- Зависимостей может быть множество
- Каждый таск выполняется только 1 раз



ЗАВИСИМОСТИ

build.cake x

```

1 run task | debug task
2 Task("Clean")
3     .Does(() => {
4         Information("Cleaning...");
5     });
6
7 run task | debug task
8 Task("Checkout")
9     .IsDependentOn("Clean")
10    .Does(() => {
11        Information("Checkouting...");
12    });
13
14 run task | debug task
15 Task("Compile")
16     .IsDependentOn("Checkout")
17     .IsDependentOn("Clean")
18     .Does(() => {
19         Information("Compiling...");
20     });
21
22 run task | debug task
23 Task("Test")
24     .IsDependentOn("Compile")
25     .Does(() => {
26         Information("Testing...");
27     });
28
29 RunTarget("Test");

```

TERMINAL

1: zsh

```

→ cake-examples#2 ./build.sh
Feeds used:
/Users/a.durnev/.nuget/packages/
https://api.nuget.org/v3/index.json
https://proget.x-plarium.com/nuget/plarium

All packages listed in /Users/a.durnev/Projects/tech-advice/cake-examples#2/tools/packages.config are already installed.

=====
Clean
=====
Cleaning...
=====
Checkout
=====
Checkouting...
=====
Compile
=====
Compiling...
=====
Test
=====
Testing...
=====
Task                                     Duration
-----
Clean                                     00:00:00.0056814
Checkout                                 00:00:00.0001497
Compile                                  00:00:00.0001155
Test                                     00:00:00.0001136
-----
Total:                                   00:00:00.0060602
→ cake-examples#2

```



КРИТЕРИИ

- Необходимы для контроля выполнения билд скрипта
- Критерии вычисляются на момент создания задачи
- Если нужно отложить вычисление до момента выполнения задачи – стоит использовать лямбды
- Обычно используются для изменения логики в зависимости от окружения, бранча, прочее

КРИТЕРИИ

build.cake x

```

1  var isLocalBuild = BuildSystem.IsLocalBuild;
2  var isRecoveryMode = false;
3
4  run task | debug task
5  Task("Checkout")
6  {
7      .WithCriteria(!isLocalBuild)
8      .Does(() =>
9      {
10         Information("Checkouting...");
11     });
12
13     run task | debug task
14     Task("Migration")
15     {
16         .Does(() =>
17         {
18             Information("Migration...");
19
20             isRecoveryMode = true;
21         });
22
23     run task | debug task
24     Task("Recover")
25     {
26         .WithCriteria(() => isRecoveryMode)
27         .Does(() =>
28         {
29             Information("Recovering...");
30         });
31
32     run task | debug task
33     Task("Deploy")
34     {
35         .IsDependentOn("Checkout")
36         .IsDependentOn("Migration")
37         .IsDependentOn("Recover")
38         .Does(() =>
39         {
40             Information("Deploying...");
41         });
42
43     RunTarget("Deploy");

```

TERMINAL

1: zsh

→ cake-examples#2 ./build.sh

Feeds used:

</Users/a.durnev/.nuget/packages/>
<https://api.nuget.org/v3/index.json>
<https://proget.x-plarium.com/nuget/plarium>

All packages listed in /Users/a.durnev/Projects/tech-advic
e/cake-examples#2/tools/packages.config are already instal
led.

=====

Migration

=====

Migration...

=====

Recover

=====

Recovering...

=====

Deploy

=====

Deploying...

=====

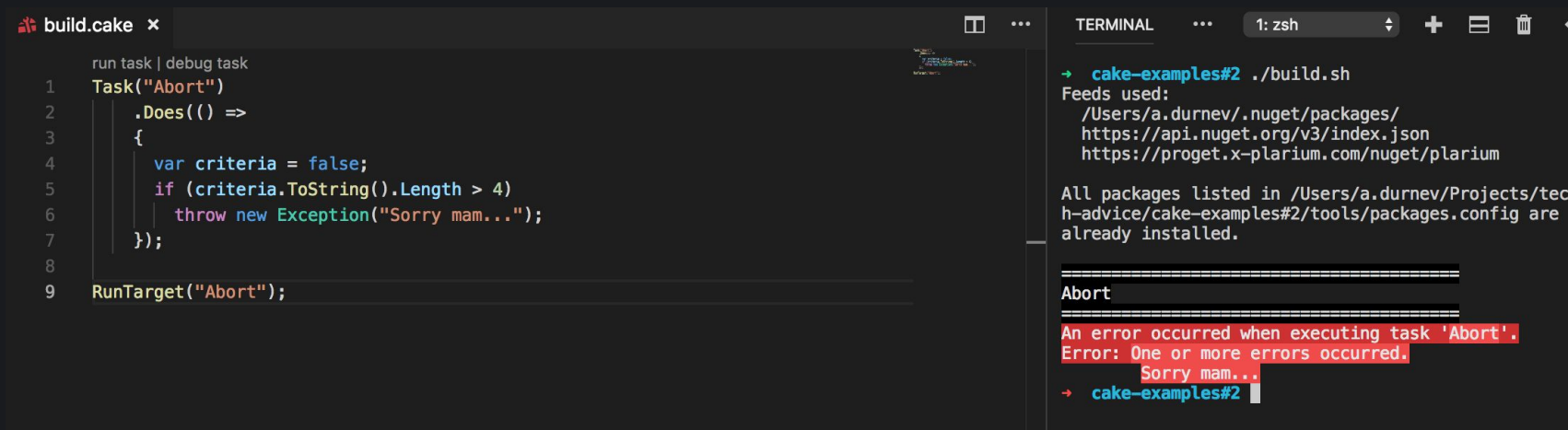
Task	Duration
Checkout	Skipped
Migration	00:00:00.0053332
Recover	00:00:00.0002034
Deploy	00:00:00.0001947
Total:	00:00:00.0057313

→ cake-examples#2

РАБОТА С ОШИБКАМИ

- Выполнение скрипта прерывается при возникновении необработанного исключения
- Для задач существует несколько вариантов обработки исключений:
 - `OnError` – колбек для обработки ошибки, продолжает выполнение
 - `ContinueOnError` – продолжает выполнение скрипта
 - `ReportError` – колбек для уведомления про исключение, останавливает выполнение скрипта

РАБОТА С ОШИБКАМИ



The screenshot shows a code editor with a file named `build.cake` and a terminal window. The code in `build.cake` defines a task named `Abort` that throws an exception with the message "Sorry mam...". The terminal shows the execution of `cake-examples#2 ./build.sh`, which results in an error message: "An error occurred when executing task 'Abort'. Error: One or more errors occurred. Sorry mam..."

```
run task | debug task
1 Task("Abort")
2     .Does(() =>
3     {
4         var criteria = false;
5         if (criteria.ToString().Length > 4)
6             throw new Exception("Sorry mam...");
7     });
8
9 RunTarget("Abort");
```

TERMINAL 1: zsh

→ cake-examples#2 ./build.sh

Feeds used:

- /Users/a.durnev/.nuget/packages/
- https://api.nuget.org/v3/index.json
- https://proget.x-plarium.com/nuget/plarium

All packages listed in /Users/a.durnev/Projects/tech-advice/cake-examples#2/tools/packages.config are already installed.

=====
Abort
=====

An error occurred when executing task 'Abort'.
Error: One or more errors occurred.
Sorry mam...

→ cake-examples#2

РАБОТА С ОШИБКАМИ

build.cake x

```
run task | debug task
1 Task("ErrorProcessing")
2     .Does(() =>
3     {
4         var criteria = false;
5         if (criteria.ToString().Length > 4)
6             throw new Exception("Sorry mam...");
7     })
8     .OnError(exception =>
9     {
10         Information("It's ok.");
11     });
12
13 RunTarget("ErrorProcessing");
```

TERMINAL

1: zsh

```
→ cake-examples#2 ./build.sh
Feeds used:
/Users/a.durnev/.nuget/packages/
https://api.nuget.org/v3/index.json
https://proget.x-plarium.com/nuget/plarium

All packages listed in /Users/a.durnev/Projects/tec
h-advice/cake-examples#2/tools/packages.config are
already installed.

=====
ErrorProcessing
=====

An error occurred when executing task 'ErrorProcess
ing'.
It's ok.

Task                                     Duration
-----
ErrorProcessing                         00:00:00.0093932
Total:                                  00:00:00.0093932
→ cake-examples#2
```



РАБОТА С ОШИБКАМИ

build.cake x

```
run task | debug task
1 Task("ContinueOnError")
2   .ContinueOnError()
3   .Does(() =>
4     {
5       var criteria = false;
6       if (criteria.ToString().Length > 4)
7         throw new Exception("Sorry mam... mam?");
8     });
9
10 RunTarget("ContinueOnError");
```

TERMINAL

1: zsh

→ cake-examples#2 ./build.sh

Feeds used:

/Users/a.durnev/.nuget/packages/
https://api.nuget.org/v3/index.json
https://proget.x-plarium.com/nuget/plarium

All packages listed in /Users/a.durnev/Projects/tec
h-advice/cake-examples#2/tools/packages.config are
already installed.

=====

ContinueOnError

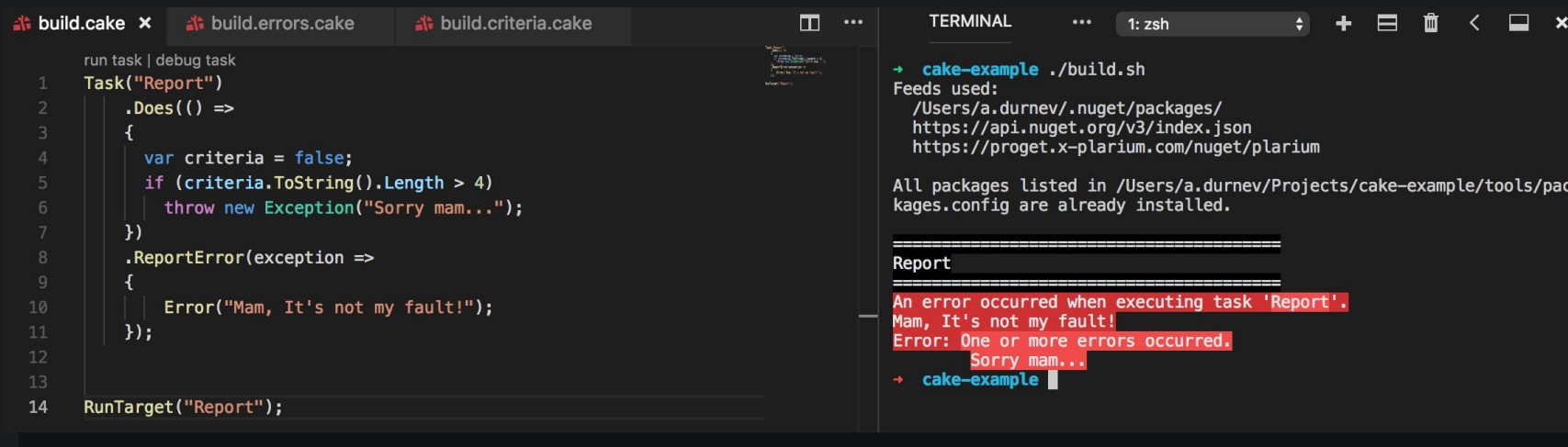
=====

An error occurred when executing task 'ContinueOnEr
ror'.

Task	Duration
ContinueOnError	00:00:00.0076059
Total:	00:00:00.0076059

→ cake-examples#2

РАБОТА С ОШИБКАМИ



The screenshot displays a code editor with three tabs: `build.cake`, `build.errors.cake`, and `build.criteria.cake`. The `build.cake` tab is active, showing a C# script with a `Task("Report")` that throws an exception if a criteria is not met. The terminal on the right shows the execution of `cake-example ./build.sh`, which lists feeds and reports an error: "An error occurred when executing task 'Report'. Mam, It's not my fault! Error: One or more errors occurred. Sorry mam..."

```
run task | debug task
1 Task("Report")
2     .Does(() =>
3     {
4         var criteria = false;
5         if (criteria.ToString().Length > 4)
6             throw new Exception("Sorry mam...");
7     })
8     .ReportError(exception =>
9     {
10         Error("Mam, It's not my fault!");
11     });
12
13
14 RunTarget("Report");
```

TERMINAL 1: zsh

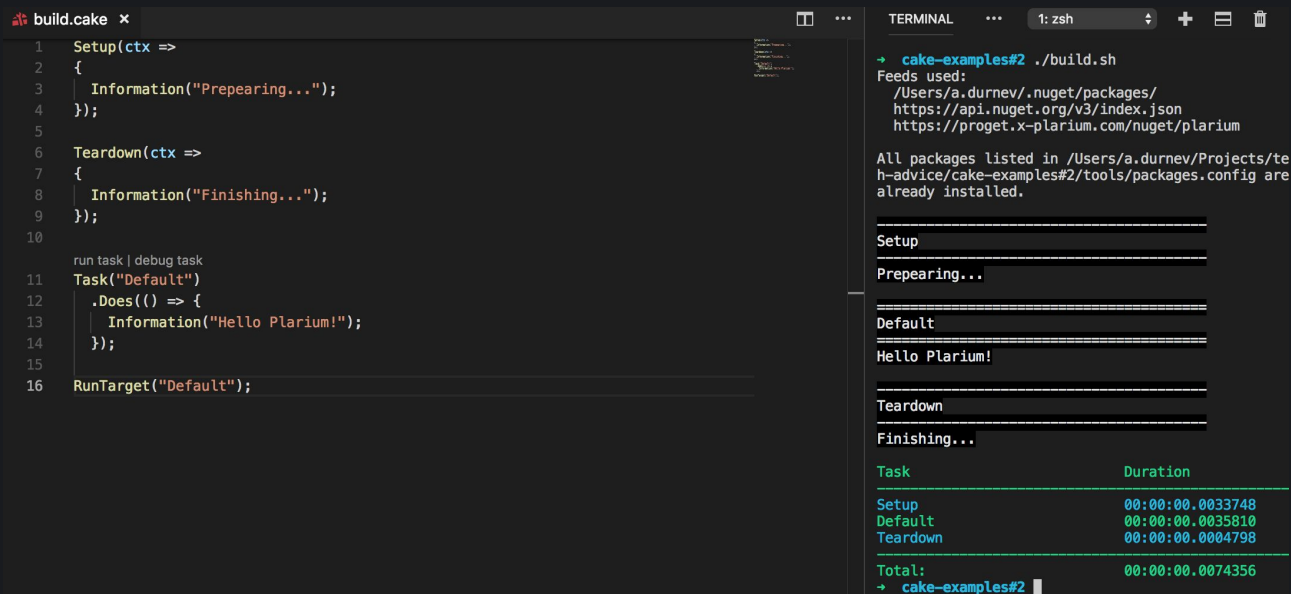
```
→ cake-example ./build.sh
Feeds used:
/Users/a.durnev/.nuget/packages/
https://api.nuget.org/v3/index.json
https://proget.x-plarium.com/nuget/plarium

All packages listed in /Users/a.durnev/Projects/cake-example/tools/packages.config are already installed.

=====
Report
=====
An error occurred when executing task 'Report'.
Mam, It's not my fault!
Error: One or more errors occurred.
    Sorry mam...
→ cake-example
```

PRE/POST TASKS

- **Setup** – существует для выполнения перед первым taskом
- **Teardown** – существует для выполнения после последнего taskа



The screenshot shows a code editor with a file named `build.cake` and a terminal window. The code defines `Setup` and `Teardown` tasks, a `Default` task, and runs the `Default` task. The terminal output shows the execution of these tasks and a summary table.

```
1 Setup(ctx =>
2 {
3     Information("Preparing...");
4 });
5
6 Teardown(ctx =>
7 {
8     Information("Finishing...");
9 });
10
11 run task | debug task
12 Task("Default")
13     .Does(() => {
14         Information("Hello Plarium!");
15     });
16 RunTarget("Default");
```

Terminal Output:

```
cake-examples#2 ./build.sh
Feeds used:
/Users/a.durnev/.nuget/packages/
https://api.nuget.org/v3/index.json
https://proget.x-plarium.com/nuget/plarium

All packages listed in /Users/a.durnev/Projects/te
h-advice/cake-examples#2/tools/packages.config are
already installed.

=====
Setup
=====
Preparing...

=====
Default
=====
Hello Plarium!

=====
Teardown
=====
Finishing...

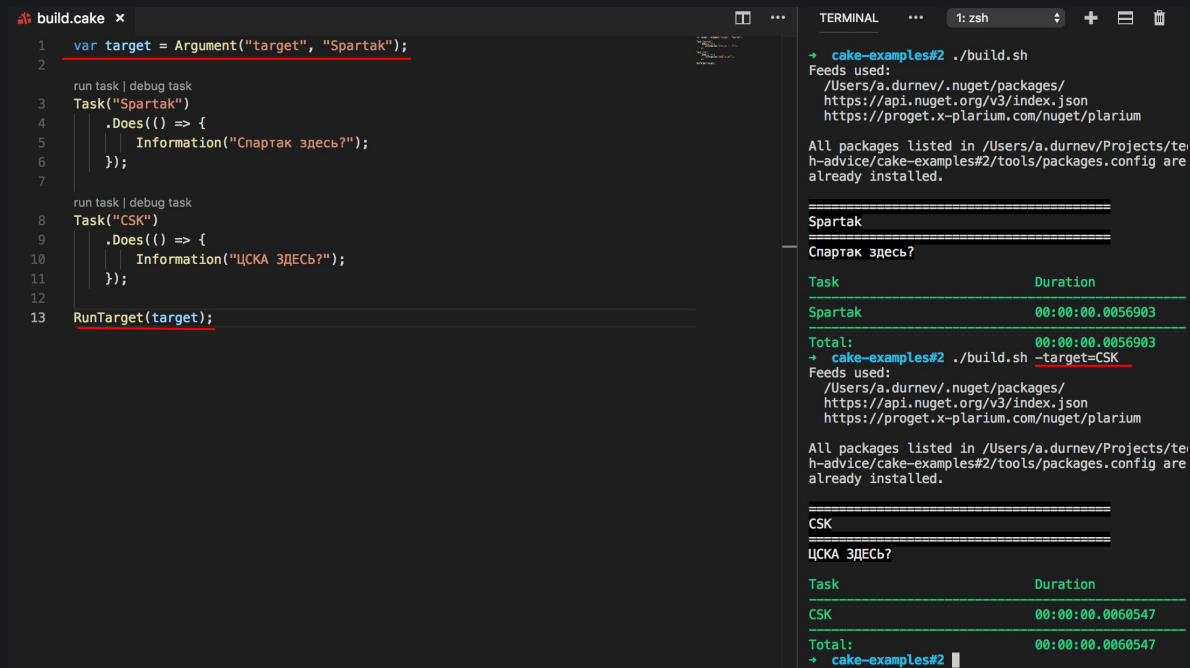
=====
Task                                     Duration
-----
Setup                                   00:00:00.0033748
Default                                00:00:00.0035810
Teardown                               00:00:00.0004798
-----
Total:                                  00:00:00.0074356
cake-examples#2
```


АРГУМЕНТЫ

- Основной способ передачи настроек в скрипт
- Конвертируются из строкового представления в нужный вам тип
- Также можно обращаться к переменным окружения

АРГУМЕНТЫ

- Аргумент `target` – стандартный подход к вызову нужного вам таска
- Старайтесь указывать имя таска по умолчанию, чтоб упростить вызов скрипта



The screenshot shows a code editor with a file named `build.cake` and a terminal window. The code defines two tasks, `Spartak` and `CSK`, and uses `RunTarget(target);` to execute them. The terminal output shows the execution of `cake-examples#2 ./build.sh`, displaying package feeds, installed packages, and a table of task durations.

```
1 var target = Argument("target", "Spartak");
2
3 run task | debug task
4 Task("Spartak")
5     .Does(() => {
6         Information("Спартак здесь?");
7     });
8
9 run task | debug task
10 Task("CSK")
11     .Does(() => {
12         Information("ЦСКА ЗДЕСЬ?");
13     });
14
15 RunTarget(target);
```

TERMINAL

```
cake-examples#2 ./build.sh
Feeds used:
/Users/a.durnev/.nuget/packages/
https://api.nuget.org/v3/index.json
https://proget.x-plarium.com/nuget/plarium

All packages listed in /Users/a.durnev/Projects/te
h-advice/cake-examples#2/tools/packages.config are
already installed.

=====
Spartak
=====
Спартак здесь?

Task                                     Duration
-----
Spartak                                  00:00:00.0056903

Total:                                   00:00:00.0056903
+ cake-examples#2 ./build.sh -target=CSK
Feeds used:
/Users/a.durnev/.nuget/packages/
https://api.nuget.org/v3/index.json
https://proget.x-plarium.com/nuget/plarium

All packages listed in /Users/a.durnev/Projects/te
h-advice/cake-examples#2/tools/packages.config are
already installed.

=====
CSK
=====
ЦСКА ЗДЕСЬ?

Task                                     Duration
-----
CSK                                      00:00:00.0060547

Total:                                   00:00:00.0060547
+ cake-examples#2
```

АРГУМЕНТЫ

build.cake x

```
1 var target = Argument("target", "Run");
2
3 var word = Argument("word", string.Empty);
4 bool isGodExist = Argument<bool>("god");
5
6 run task | debug task
7 Task("Run")
8     .Does(() => {
9         if (isGodExist)
10             Information($"And God said, {word} and there was {word}");
11     });
12 RunTarget(target);
```

TERMINAL

1: zsh

```
→ cake-examples#2 ./build.sh --god=True --word=Light
Feeds used:
/Users/a.durnev/.nuget/packages/
https://api.nuget.org/v3/index.json
https://proget.x-plarium.com/nuget/plarium

All packages listed in /Users/a.durnev/Projects/tech-advice/cake-examples#2/tools/packages.config are already installed.

=====
Run
=====
And God said, Light and there was Light

Task                                     Duration
-----
Run                                     00:00:00.0056641
Total:                                  00:00:00.0056641
→ cake-examples#2
```



АРГУМЕНТЫ

build.cake x

```
1 var target = Argument("target", "Run");
2
3 var word = Argument("word", string.Empty);
4 bool isGodExist = Argument<bool>("god");
5
6 run task | debug task
7 Task("Run")
8     .Does(() => {
9         if (isGodExist)
10             Information($"And God said, {word} and there was {word}");
11     });
12 RunTarget(target);
```

TERMINAL

1: zsh

→ cake-examples#2 ./build.sh --god=True

Feeds used:

/Users/a.durnev/.nuget/packages/
https://api.nuget.org/v3/index.json
https://proget.x-plarium.com/nuget/plarium

All packages listed in /Users/a.durnev/Projects/tech-advice/cake-examples#2/tools/packages.config are already installed.

Run

And God said, and there was

Task	Duration
------	----------

Run	00:00:00.0058531
-----	------------------

Total: 00:00:00.0058531

→ cake-examples#2 ./build.sh --word=Light

Feeds used:

/Users/a.durnev/.nuget/packages/
https://api.nuget.org/v3/index.json
https://proget.x-plarium.com/nuget/plarium

All packages listed in /Users/a.durnev/Projects/tech-advice/cake-examples#2/tools/packages.config are already installed.

Error: One or more errors occurred.
Argument 'god' was not set.

→ cake-examples#2



АРГУМЕНТЫ

build.cake x

```
1 var target = Argument("target", "Run");
2
3 var word = EnvironmentVariable("word") ?? "Light";
4 bool isGodExist() => Argument<bool>("god");
5
6 run task | debug task
7 Task("Clear")
8 | .Does(() => {
9 |     Information("Clearing...");
10 | });
11
12 run task | debug task
13 Task("Run")
14 | .Does(() => {
15 |     if (isGodExist())
16 |         Information($"And God said, {word} and there was {word}");
17 | });
18
19 RunTarget(target);
```

TERMINAL

1: zsh

→ cake-examples#2 ./build.sh -target=Clear

Feeds used:

/Users/a.durnev/.nuget/packages/
https://api.nuget.org/v3/index.json
https://proget.x-plarium.com/nuget/plarium

All packages listed in /Users/a.durnev/Projects/tech-advice/cake-examples#2/tools/packages.config are already installed.

Clear

Clearing...

Task	Duration
Clear	00:00:00.0054885
Total:	00:00:00.0054885

→ cake-examples#2



ДИРЕКТИВЫ ПРЕПРОЦЕССОРА

- Директивы используются для загрузки в скрипт различных расширений, скриптов и nuGet библиотек
- Можно использовать пакеты как с nuGet сервера, так и локальные скрипты или либы

ДИРЕКТИВЫ ПРЕПРОЦЕССОРА

build.cake x

```

1 #load nuget: [PlariumFeed]?package=Advisory.Cake
2 #tool nuget: ?package=NUnit.ConsoleRunner
3
4 #load "MessageProvider.cake"
5
6 var target = Argument("target", "Fight");
7
8 run task | debug task
9 Task("Fight")
10 {
11     .Does(() => {
12         var provider = new MessageProvider();
13
14         var message = new StringBuilder();
15         message.AppendLine($"- {provider.KansiroSays()}");
16         message.AppendLine($"- {provider.EnemyAnswer()}");
17
18         Information(message.ToString());
19     });
20 }
21 RunTarget(target);

```

MessageProvider.cake x

```

1 class MessageProvider
2 {
3     public string KansiroSays()
4     {
5         return "Omae wa mou shindeiru";
6     }
7
8     public string EnemyAnswer()
9     {
10         return "Nani?!?!?!";
11     }
12 }

```

1: zsh

→ cake-examples#2 ./build.sh

Feeds used:

/Users/a.durnev/.nuget/packages/

https://api.nuget.org/v3/index.json

https://proget.x-plarium.com/nuget/plarium

All packages listed in /Users/a.durnev/Projects/tech-advice/cake-examples#2/tools/packages.config are already installed.

The assembly 'Cake.Slack, Version=0.12.0.0, Culture=neutral, PublicKeyToken=null' is referencing an older version of Cake.Core (0.26.0). For best compatibility it should target Cake.Core version 0.28.0.

=====

Fight

=====

- Omae wa mou shindeiru

- Nani?!?!?!

=====

Task	Duration
Fight	00:00:00.0034192
Total:	00:00:00.0034192

→ cake-examples#2



АРГУМЕНТЫ КОМАНДНОЙ СТРОКИ

- Существуют для передачи аргументов в билд скрипт
- Формат передачи аргументов в sh и ps1 bootstrap скрипты – могут отличаться
- Формат передачи аргументов также отличается в разных версиях bootstrap скриптов
- Из полезных аргументов можно выделить:
 - `-dryrun` – выполняет сборку скрипта и выводит очередь выполнения задач
 - `-verbosity` – задает минимальный уровень логов (quiet, minimal, normal, verbose, diagnostic)
- Примеры вызова скриптов:
 - `./build.sh --script=custom-build.cake --target="Integration" --verbosity=normal`
 - `.\build.ps1 -Target Deployment --nuGetApiKey=proget:xXxXXxXxXXxX`

ИНТЕГРАЦИЯ С BAMBOO – WIN

Source Code Checkout
Checkout Default Repository

Script
build.ps1

Final tasks Are always executed even if a previous task fails
Drag tasks here to make them final

Add task

Script configuration

Task description
build.ps1

☐ Disable this task

Interpreter
Windows PowerShell

Run your script with Windows PowerShell.

Script location
File

Script file *
./build.ps1

Argument
-Target Deployment --NuGetApiKey=\${bamboo.NuGetApiKey}

Environment variables

Working subdirectory

Save Cancel

ИНТЕГРАЦИЯ С BAMBOO – MAC

Source Code Checkout
Checkout Default Repository

Script
Cake

Final tasks Are always executed even if a previous task fails

Drag tasks here to make them final

Add task

Script configuration

Task description

☐ Disable this task

Interpreter

Run your script with /bin/sh or cmd.exe

Script location

Script file*

Argument

Environment variables

Working subdirectory

Save Cancel

ПРИМЕРЫ ИСПОЛЬЗОВАНИЯ **CAKE** В НАШИХ ПРОЕКТАХ



THANK YOU



TAKE THE WORLD