

# Методические указания по теме “workflow #1”

## Содержание

1. [Terminal](#)
2. [Npm](#)
3. [Bower](#)
4. [Git](#)
5. [BrowserSynk](#)
6. [Gulp](#)
7. [ПОШАГОВОЕ РУКОВОДСТВО](#)



# 1. Terminal

## 1.1. Ссылки

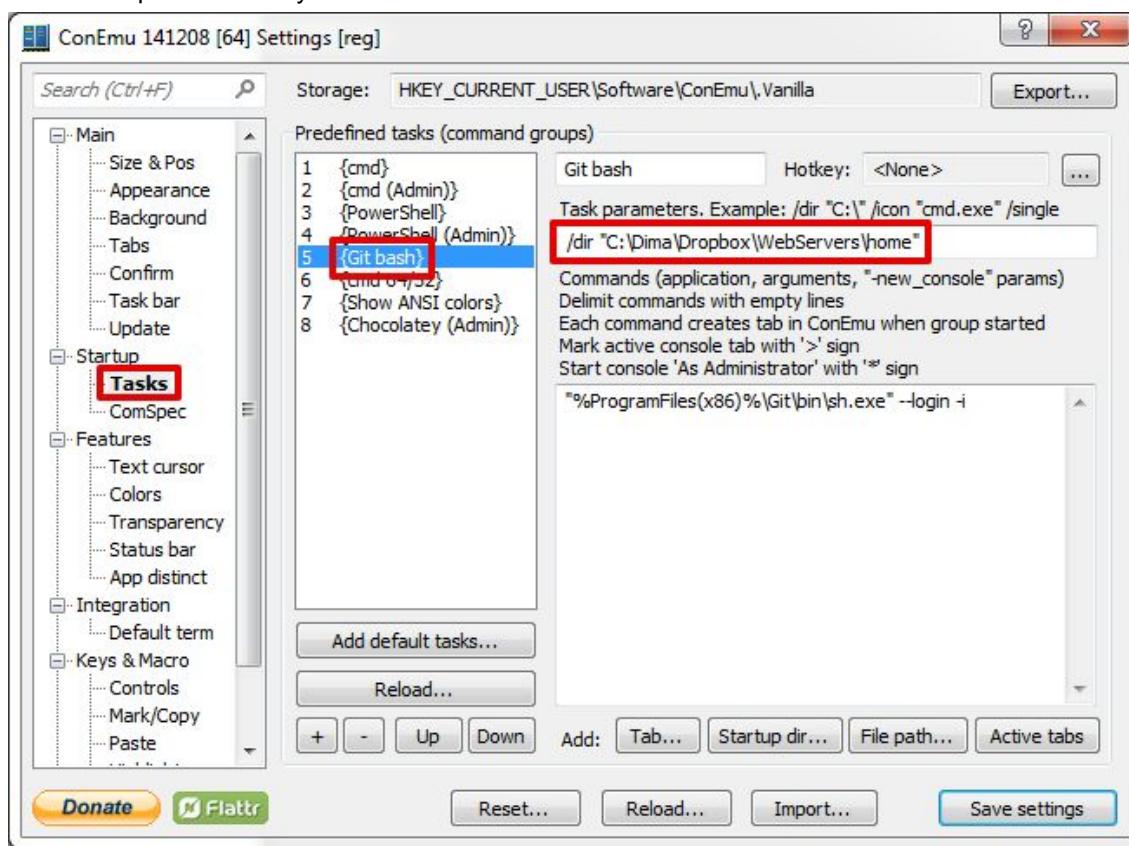
- 1.1.1. <http://www.youtube.com/watch?v=x0hw8IIIzKY> - подробный урок от Ковальчука Дмитрия по ConEmu - установка, настройка

## 1.2. Рекомендуемые консоли

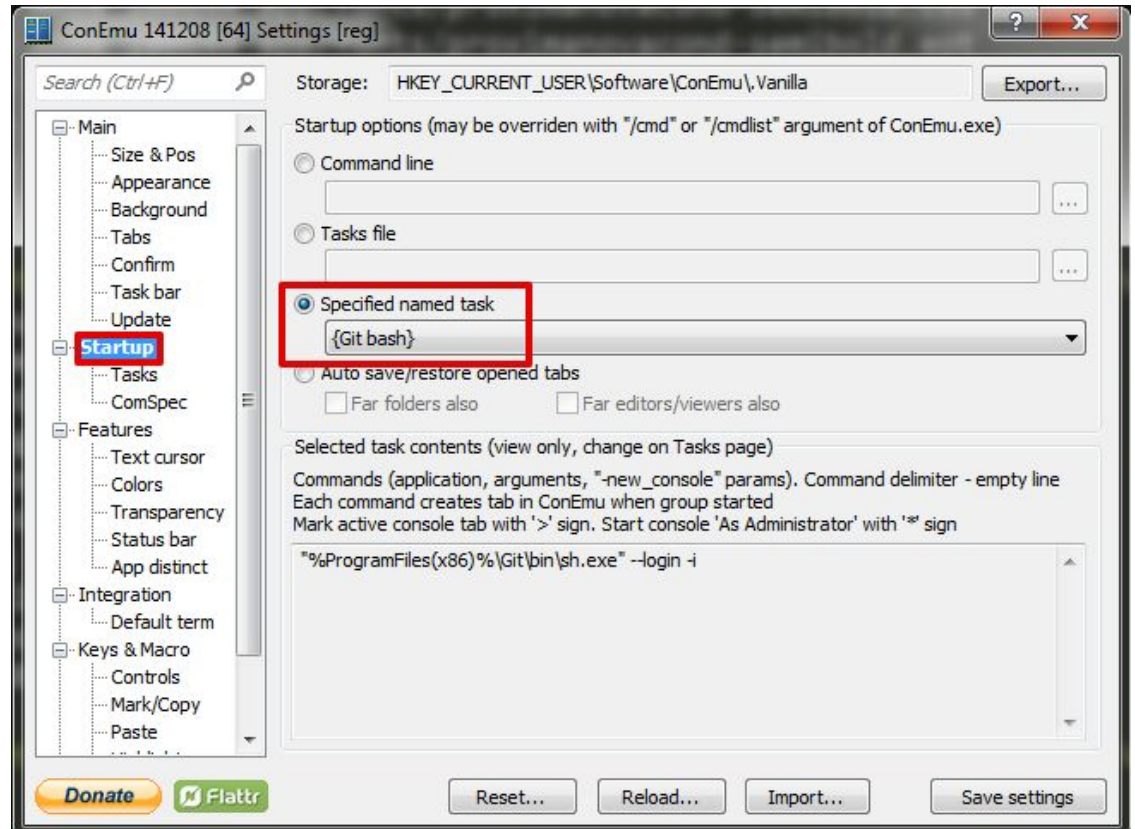
- 1.2.1. Conemu - <http://www.conemu.ru/>
- 1.2.2. Cmdr - <http://gooseberrycreative.com/cmdr/>
- 1.2.3. item 2 - <http://item2.com/> (только mac OS)
- 1.2.4. git bash - <http://git-scm.com/downloads>
- 1.2.5. Console2 - <http://sourceforge.net/projects/console/>

## 1.3. Настройка сопему

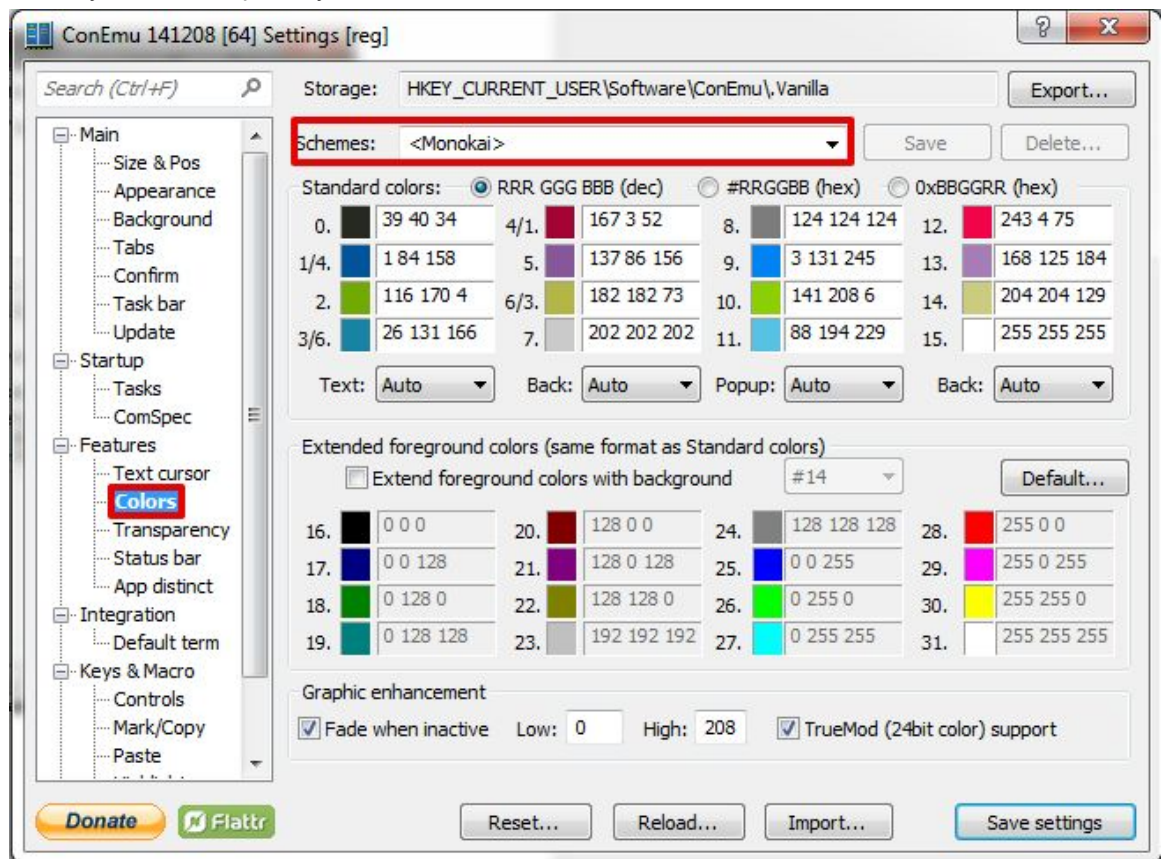
- 1.3.1. CTRL + ALT + P - заходим в настройки
- 1.3.2. Настройка задачи. Переходим в Git Bash и указываем путь в каталог, который будет открываться по-умолчанию:



1.3.3. Затем выбираем task по-умолчанию:



1.3.4. Тему можно выбрать тут:



## 1.4. Полезные команды

### Основные команды

pwd	- показать текущий каталог
cd	- переход в заданную папку
cd -	- перейти в предыдущую директорию
ls	- показывает список файлов текущей папки
ls -f	- покажет также .htaccess
cp	- копирование файлов/папок
mv	- переименовать или переместить файлы/папки
rm	- удаление файлов/папок
rm -r	- удаление папки и её содержимого
mkdir	- создать папку
clear	- очистить консоль
touch	- создать файл

### Подробнее

cd /home	- перейти в директорию '/home'
cd ..	- перейти в директорию уровнем выше
cd ../../	- перейти в директорию двумя уровнями выше
cd	- перейти в домашнюю директорию
cd ~user	- перейти в домашнюю директорию пользователя user
cd -	- перейти в директорию, в которой находились до перехода в текущую директорию
pwd	- показать текущую директорию
ls	- отобразить содержимое текущей директории
ls -l	- покажет, кто создал, дату создания и пр.
ls -a	- показать скрытые файлы и директории в текущей директории
mkdir dir1	- создать директорию с именем 'dir1'
mkdir dir1 dir2	- создать две директории одновременно
mkdir -p /tmp/dir1/dir2	- создать дерево директорий
rm -f file1	- удалить файл с именем 'file1'
rmdir dir1	- удалить директорию с именем 'dir1'
mv dir1 new_dir	- переименовать или переместить файл или директорию
cp file1 file2	- скопировать файл file1 в файл file2
cp dir/* .	- копировать все файлы директории dir в текущую директорию
cp -a /tmp/dir1	- копировать директорию dir1 со всем содержимым в текущую директорию
cp -a dir1 dir2	- копировать директорию dir1 в директорию dir2

### Shortcuts

ctrl + u	удалить от курсора - до начала строки
ctrl + k	удалить от курсора - до конца строки
ctrl + w	удалить от курсора - до начала слова
ctrl + a	перейти к началу строки



ctrl + e            перейти к концу строки

### **Создание нескольких вложенных файлов**

touch css/{main.css,buttons.css} js/{main.js,plugins.js}            - без пробелов внутри {}

### **Создать папку “www” и перейти в неё**

mkdir www && cd www

## 2. Npm

### 2.1. Установка

2.1.1. Качаем **node.js + npm** с официального сайта: <https://nodejs.org/download/>

2.1.2. Плагины можно искать тут <https://www.npmjs.com/>

### 2.2. Как работать

2.2.1. Создаём файл **package.json**, содержащий

```
{ }
```

2.2.2. Установка пакетов

```
npm install --save-dev gulp
```



## 3. Bower

### 3.1. Полезные ссылки

3.1.1. **Обязательно** посмотреть урок: <http://www.youtube.com/watch?v=jl3-rYqnvqU>

3.1.2. При возникновении любых вопросов, читаем статью:  
<http://loftblog.ru/2015/06/18/bower-podrobnoe-rukovodstvo-1/#more-3283>

3.1.3. Официальный сайт: <http://bower.io/>

### 3.2. Установка

```
npm install -g bower
```

### 3.3. Как изменить директорию, куда по-умолчанию будут устанавливаться пакеты.

3.3.1. Создаём файл **.bowerrc**, содержащий

```
{  
    "directory": "app/bower"  
}
```

### 3.4. Как работать

3.4.1. Создаём файл **bower.json**, содержащий

```
{  
    "name" : "dz1"  
}
```

3.4.2. Установка пакетов

```
bower install --save jquery#1.11
```



## 4. Git

4.1. **Алиасы** <http://githowto.com/ru/aliases>

4.2. **Возможные ошибки**

4.2.1. переводы строк в разных системах <http://jenyay.net/Git/Autocrlf>

Проблема с переносом строк

```
git config core.autocrlf false
```

4.3. **Частые команды**

```
git init
```

```
git add .
```

```
git status
```

```
git commit -m "обычный комит"
```

```
git commit -am "комит, если какой-либо файл был удалён"
```

```
git reset --hard
```

4.4. **Github**

4.4.1. **Как клонировать репозиторий на свой компьютер?**

```
git clone https://github.com/__username__/__reponame__.git __pathname__
```

4.4.2. **Как стягивать последние изменения с репозитория?**

```
git pull origin master
```

4.4.3. **Как отправить файлы на Github?**

**Первый раз:**

```
git remote add origin https://github.com/__username__/__reponame__.git  
git push origin master
```

**Каждый последующий раз:**

```
git push origin master
```

4.5. **.gitignore**

```
node_modules  
app/bower_components  
dist
```





#### 4.6. Алиасы

Добавить алиас можно двумя путями:

1. Набрать в консоли команду `git config --global alias.алиас "строка команды"`

Пример: `git config --global alias.st "status"`

2. В файле конфигурации git `.gitconfig` (хранится в папке пользователя системы) добавить секцию `[alias]` и ниже нее перечислить алиасы в формате `alias = команда [опции]`

```
[alias]
  st = status
  ci = commit
  co = checkout
  br = branch
  unstage = reset HEAD --
  last = log -1 HEAD
```



## 5. BrowserSync

5.1. Ссылка <http://www.browsersync.io/>

5.2. Установка

```
npm install -g browser-sync
```

5.3. Запуск

```
browser-sync start --server app --files "app/**/*.html"
```

`--server app` - указывает на то, в какой папке лежит наш index.html

`--files "app/**/*.html"` - указывает на то, за какими файлами следить

## 6. Gulp

### 6.1. Полезные ссылки

- 6.1.1. Официальный сайт: <http://gulpjs.com/>
- 6.1.2. Поиск плагинов: <http://gulpjs.com/plugins/>
- 6.1.3. Обязательно ознакомиться с курсом по gulp:  
<http://www.youtube.com/playlist?list=PLY4rE9dstrJwXCz1utct9b6Vub9VWQoKo>

### 6.2. Установка

1. Установка Gulp глобально - обязательно сделать это в первый раз

```
$ npm install --global gulp
```

2. Установка Gulp локально в папку нашего проекта + записываем информацию в package.json

```
$ npm install --save-dev gulp
```

3. Создаём gulpfile.js в корне проекта. В нём пишем необходимые нам задачи, например:

```
var gulp = require('gulp');  
  
gulp.task('default', function() {  
    ... задача по-умолчанию  
});
```

4. Запуск Gulp

```
gulp
```

## 7. ПОШАГОВОЕ РУКОВОДСТВО

как сделать, чтобы работало точно так же, как и на вебинаре

### 1) Внимание!

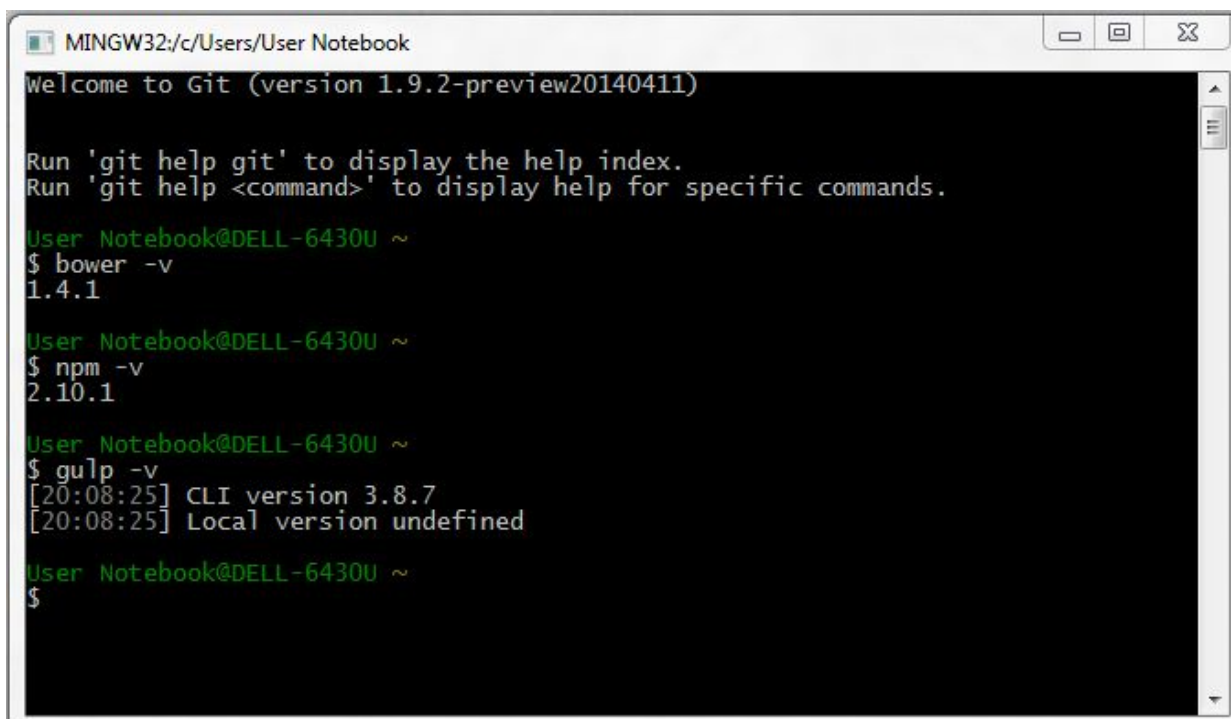
- а) Данное руководство гарантированно будет работать в консоли **git bash**. Если вы работаете в иной консоли и у вас что-то не получается, прежде всего установите именно **git bash** и попробуйте в ней.
- б) Убедитесь, что у вас установлены bower, npm и gulp. Для этого используйте следующие команды:

```
bower -v
```

```
npm -v
```

```
gulp -v
```

В консоли **git bash** вы увидите примерно следующее



```
MINGW32/c/Users/User Notebook
Welcome to Git (version 1.9.2-preview20140411)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

User Notebook@DELL-6430U ~
$ bower -v
1.4.1

User Notebook@DELL-6430U ~
$ npm -v
2.10.1

User Notebook@DELL-6430U ~
$ gulp -v
[20:08:25] CLI version 3.8.7
[20:08:25] Local version undefined

User Notebook@DELL-6430U ~
$
```

### 2) Создаём структуру будущего сайта

```
mkdir workflow-les1
```

```
cd !$
```

```
mkdir -p app/{css,js}
```

```
touch app/{css/main.css,js/main.js,index.html}
```



3) Для работы с bower подготовим необходимые файлы

```
touch bower.json .bowerrc
```

4) В bower.json напишем

```
{  
  "name" : "site",  
  "version" : "0.1.0"  
}
```

5) В .bowerrc укажем, куда именно должны устанавливаться пакеты

```
{  
  "directory": "app/bower"  
}
```

6) Установим jquery версии 1.8 и normalize.css

```
bower i --save jquery#1.8 normalize.css
```

7) Заполним index.html файл

```
<!DOCTYPE html>  
<html>  
<head>  
  <meta charset="UTF-8">  
  <title>Workflow</title>  
  <link rel="stylesheet" href="bower/normalize.css/normalize.css">  
  <link rel="stylesheet" href="css/main.css">  
</head>  
<body>  
  <h1>Workflow - часть 1</h1>  
  <p>Просто текст, который мы будем изменять при помощи livereload</p>  
  <script src="bower/jquery/jquery.js"></script>  
  <script src="js/main.js"></script>  
</body>  
</html>
```



8) Заполним main.css файл

```
h1{
    color: green;
}
p{
    color: blue;
}
```

9) Заполним main.js файл

```
$(function() {
    console.log('Файл main.js успешно загружен')
});
```

10) Для работы с прт нужно создать файл конфигурации

```
touch package.json
```

11) В package.json напишем:

```
{
    "name" : "site",
    "version" : "0.1.0",
    "private": true
}
```

12) Установим теперь gulp и browserSync локально

13) Для работы с gulp нужно создать файл конфигурации

```
touch gulpfile.js
```

14) Заполним файл gulpfile.js

```
var gulp = require("gulp"),
    browserSync = require('browser-sync');

gulp.task('server', function () {
    browserSync({
        port: 9000,
        server: {
            baseDir: 'app'
        }
    });
});
```



```
});

gulp.task('watch', function () {
  gulp.watch([
    'app/*.html',
    'app/js/**/*.js',
    'app/css/**/*.css'
  ]).on('change', browserSync.reload);
});

gulp.task('default', ['server', 'watch']);
```

15) Тестируем browserSync. Для этого в консоли набираем

## gulp

После чего автоматически открывается браузер и мы можем попробовать:

1. Поменять заголовок и текст параграфа в index.html
2. Поменять цвет текста в main.css
3. Поменять сообщение в консоли в main.js

При любых изменениях этих файлов браузер должен перезагружаться автоматически.