semenchenkov.github.io

Gulpfile.js

Front-end сборка проекта с помощью GulpJS

Node.js

Download

Git

Download for Windows
* 2.run for Win
* 3.

RubyInstaller

Download

СОЗДАНИЕ ИСХОДНИКОВ:

$ mkdir www Создание директории

$ cd www Переход в дир.

$ mkdir app/{css,js} –p

$ cd app

$ touch js/{main.js,plugins.js} Создание ф-ов

css/{style1.css,style2.css} index.html

$ mkdir scss/{_utilities,layout,base,modules} -p

$ cd scss

$ touch

_utilities/{_index.scss,_typography.scss,_colors.scss,

clearfix.scss}

layuout/{_index.scss,_footer.scss,_header.scss,_cont

ainer.scss} base/{_base.scss} index.scss

$ cd .. Переход на ур. выше в app

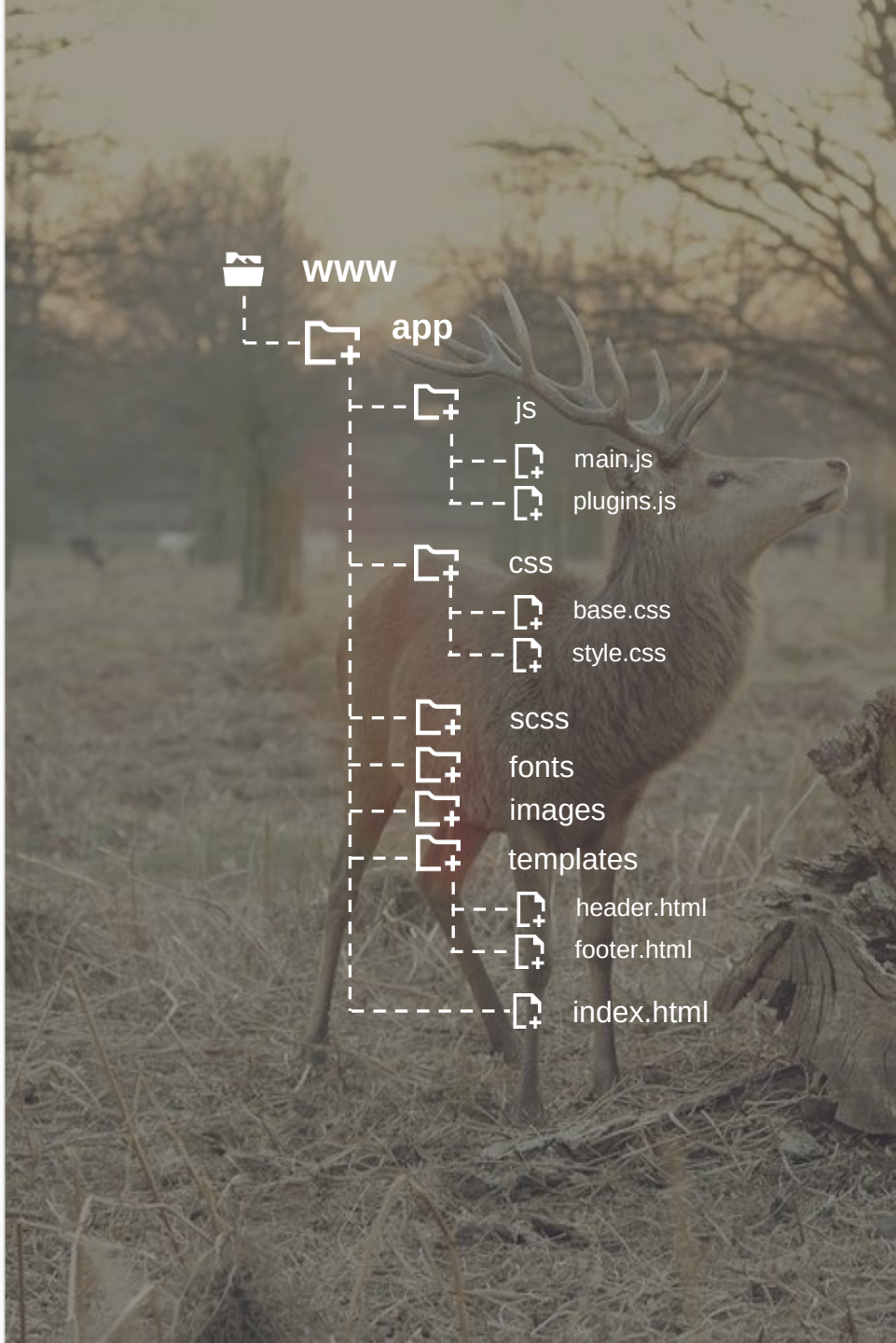
$ cd .. Переход на ур. выше в www

$ ls Просмотр файлов

rm -r dir – удаление дир. с содержимым

cd !$ - переход во вновь созд. дир или ф-л

ключ **–p** созд. и самой дир.



ФАЙЛЫ ЗАВИСИМОСТЕЙ:

$ touch .bowerrc .gitignore gulpfile.js

.gitignore - Список ф-ов игнорируемых в GIT

.bowerrc файл настройки Bower с помощью JSON

gulpfile.js файл настроек для сборки проекта

$ npm init

- Создает ф-л с глобальными настройками проекта **package.json**

$ bower init

- Создает ф-л с настройками пакетного менеджера Bower **bower.json**

УСТАНОВКА ПЛАГИНОВ:

Установка самого gulp:

$ npm i –g gulp - глобально

$ npm i --save-dev gulp - для разработчика (-D)

Локальный dev сервер:

$ npm i –g browser-sync

$ npm i --save-dev browser-sync

Наблюдение за изменениями файлов:

$ npm i gulp-watch

Очистка:

$ npm i --save-dev gulp-clean

Парсинг\конкатенация js, css, html ф-ов:

$ npm i --save-dev gulp-useref

$ npm i --save wiredep

$ npm i --save-dev gulp-rigger

Удаление не используемых стилей:

$ npm i --save-dev gulp-uncss

Автопрефиксы:

$ npm i --save-dev gulp-autoprefixer

Минификация:

$ npm i --save-dev gulp-uglify

$ npm i --save-dev gulp-css-minify

$ npm i --save-dev gulp-imagemin

$ npm i --save-dev imagemin-pngquant

Удаление\деинсталяция:

$ rm –r node_modules

$ uninstall browser-sync

$ npm cache clean ~/

Bower:

$ npm i –g bower

$ npm update –g bower

$ bower i --save jquery

$ bower i --save jquery#1.11

$ bower i bootstrap

$ bower i --save slick-carousel

$ bower i --save slick.js

GULPFILE.JS:

// Проверка строк

```
'use strict';
```

// Плагины

```
var gulp          = require('gulp'),
    watch         = require('gulp-watch'),
    rigger        = require('gulp-rigger'),
    concat        = require('gulp-concat'),
    uncss         =  require('gulp-uncss'),
    useref        = require('gulp-useref'),
    wiredep       = require('wiredep').stream,
    clean         = require('gulp-clean'),
    uglify        = require('gulp-uglify'),
    minifyCss     = require('gulp-minify-css'),
    imagemin      = require('gulp-imagemin'),
    pngquant      = require('imagemin-pngquant'),
    autoprefixer  = require('gulp-autoprefixer'),
    browserSync = require("browser-sync"),
    reload        = browserSync.reload;
```

// Пути

```
var path = {
        app: { //Исходные
            html: 'app/*.html',
            js: 'app/js/*.js',
            style: 'app/css/*.css',
            img: 'app/images/**/*.*',
            fonts: ‘app/fonts/**/*.*'
        //Все файлы всех расш-ий из папки и
        // из вложенных каталогов
        },
        dist: { //Релиз
            html: 'dist/',
            js: 'dist/js/',
            css: 'dist/css/',
            img: 'dist/images/‘,
            fonts: ‘dist/fonts/'
        },
        watch: { //Слежка
            html: 'app/**/*.html',
            js: 'app/js/**/*.js',
            style: 'app/css/**/*.css',
            img: 'app/images/**/*.*',
            fonts: ‘app/fonts/**/*.*'
        },
        clean: './dist'
};
```

GULPFILE.JS:

```
// Работа с HTML
gulp.task('html', function(){
        var assets = useref.assets();
        gulp.src(path.app.html)
                .pipe(rigger())
                .pipe(wiredep({
                        "overrides" : {
                                "bootstrap" : {
                                        "main" : [
                                                "dist/js/bootstrap.js",
                                                "dist/css/bootstrap.css“
                                        ]
                                }
                        }
                }))
                .pipe(assets)
                .pipe(assets.restore())
                .pipe(useref())
                .pipe(gulp.dest(path.dist.html))
                .pipe(reload({stream : true}));});
```

```
// Работа с CSS
gulp.task('style', function(){
        gulp.src(path.app.css)
                .pipe(autoprefixer())
                .pipe(minifyCss())
                .pipe(gulp.dest(path.dist.css))
                .pipe(reload({stream : true}));});
```

GULPFILE.JS:

// Работа с Fonts

```
gulp.task('fonts', function() {
    gulp.src(path.app.fonts)                    // app/fonts <---
        .pipe(gulp.dest(path.dist.fonts))       // --> dist/fonts
        .pipe(reload({stream: true}));          // Перезагрузка сервера
});
```

// Работа с JS

```
gulp.task('js', function () {
            gulp.src(path.app.js)                           // app/js <---
                        .pipe(gulp.dest(path.dist.js))      // --> dist/js
                        .pipe(reload({stream: true}));      // Перезагрузка сервера
});
```

// Работа с Images

```
gulp.task('image', function () {
            gulp.src(path.app.img)                          // app/images <---
                        .pipe(imagemin({                    //Сжать
                                    progressive: true,
                                    svgoPlugins: [{removeViewBox: false}],
                                    use: [pngquant()],
                                    interlaced: true
                        }))
                        .pipe(gulp.dest(path.dist.img))     // --> dist/images
                        .pipe(reload({stream: true}));      // Перезагрузка сервера
});
```

GULPFILE.JS:

// Настройка сервера browser-sync

```
var config = {
        server: {
                baseDir: "./dist"
        },
        tunnel: true,
        host: 'localhost',
        port: 9000,
        logPrefix: "SenSeless"
};
```

// Gulp-таски

```
gulp.task('dist', [
        'html',
        'style',
        'js',
        'fonts',
        'image'
]);
```

```
gulp.task('task_1', ['pre_task_1', 'pre_task_2'], function() {
  console.log('task_1 is done');
});
// Здесь мы объявили `task_1`, который выводит в консоль сообщение `task_1 is done`
// Запускается он командой `gulp task_1`
// Но перед выполнением основного `task_1` должны выполниться задачи `['pre_task_1', 'pre_task_2']`
// Важно понимать, что 'pre_task_1' & 'pre_task_2' - выполняются асинхронно, // то есть порядок выполнения не зависит от позиции задачи в массиве, // а `task_1` стартует только после того, как отработали 2 pre-задачи - то есть синхронно
```

// Запуск сервера browser-sync

```
gulp.task('webserver', function () {
        browserSync(config);
});
```

GULPFILE.JS:

// Чистка

```
gulp.task('clean', function (cb) {
        clean(path.clean, cb);
});
```

// Слежка (после чистки)

```
gulp.task('watch', function(){
        watch([path.watch.html], function(event, cb) {
                gulp.start('html');
        });
        watch([path.watch.style], function(event, cb) {
                gulp.start('style');
        });
        watch([path.watch.js], function(event, cb) {
                gulp.start('js');
        });
        watch([path.watch.fonts], function(event, cb) {
                gulp.start('fonts');
        });
        watch([path.watch.img], function(event, cb) {
                gulp.start('image');
        });
});
```

// Задача по-умолчанию

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

INDEX.HTML

```
<!DOCTYPE html>
<html lang="ru">
<head>
        <meta charset="UTF-8">
        <title>Test-pj</title>

        <!-- build:css css/vendor.css -->
                <!-- bower:css -->
                <!-- endbower -->
        <!-- endbuild -->

        <!-- build:css css/global.css -->
                <link rel="stylesheet" href="css/base.css"></link>
                <link rel="stylesheet" href="css/style.css"></link>
        <!-- endbuild -->
</head>
<body>
        //= header.html <!-- Подключение header с помощью gulp-rigger -->
        <main> <h1>hello, world</h1>    </main>
        //= footer.html <!-- Подключение footer с помощью gulp-rigger -->

        <!-- build:js scripts/combined.js -->
                <!-- bower:js -->
                <!-- endbower -->
        <!-- endbuild -->
</body>
</html>
```

СТРУКТУРА RELISE ПРОЕКТА:

Сборка из исходников:

- wiredep достает из bower css и js ф-лы (bootstrap.css, jquery.js, ...)
- useref собирает их в общий ф-л vendor.css и vendor.js

P.S: При необходимости (либо при крайней сборке релиз ф-ов в dist) можно объеденить свои стили и скрипты

```
<!-- build:css css/global.css --> ... <!-- endbuild -->
<!-- build:js js/main.js --> ... <!-- endbuild -->
```

- rigger подключает tamplate и собирает *.html
- gulp-minify-css gulp-useref gulp-imagemin gulp-svgmin ... Минифицируют

Запуск watcher-ов «слежка», которые будут пере собирать проект при изменении исходных файлов + Запуск локального сервера

- browserSync автоматически обновляет браузер + clean очищает dist

GITLAB:

$ git init

$ git add .

$ git commit –am “add all files”

New repository gitlab

$ git remote add origin http:// …

$ git push –u origin master

$ git push –f origin master

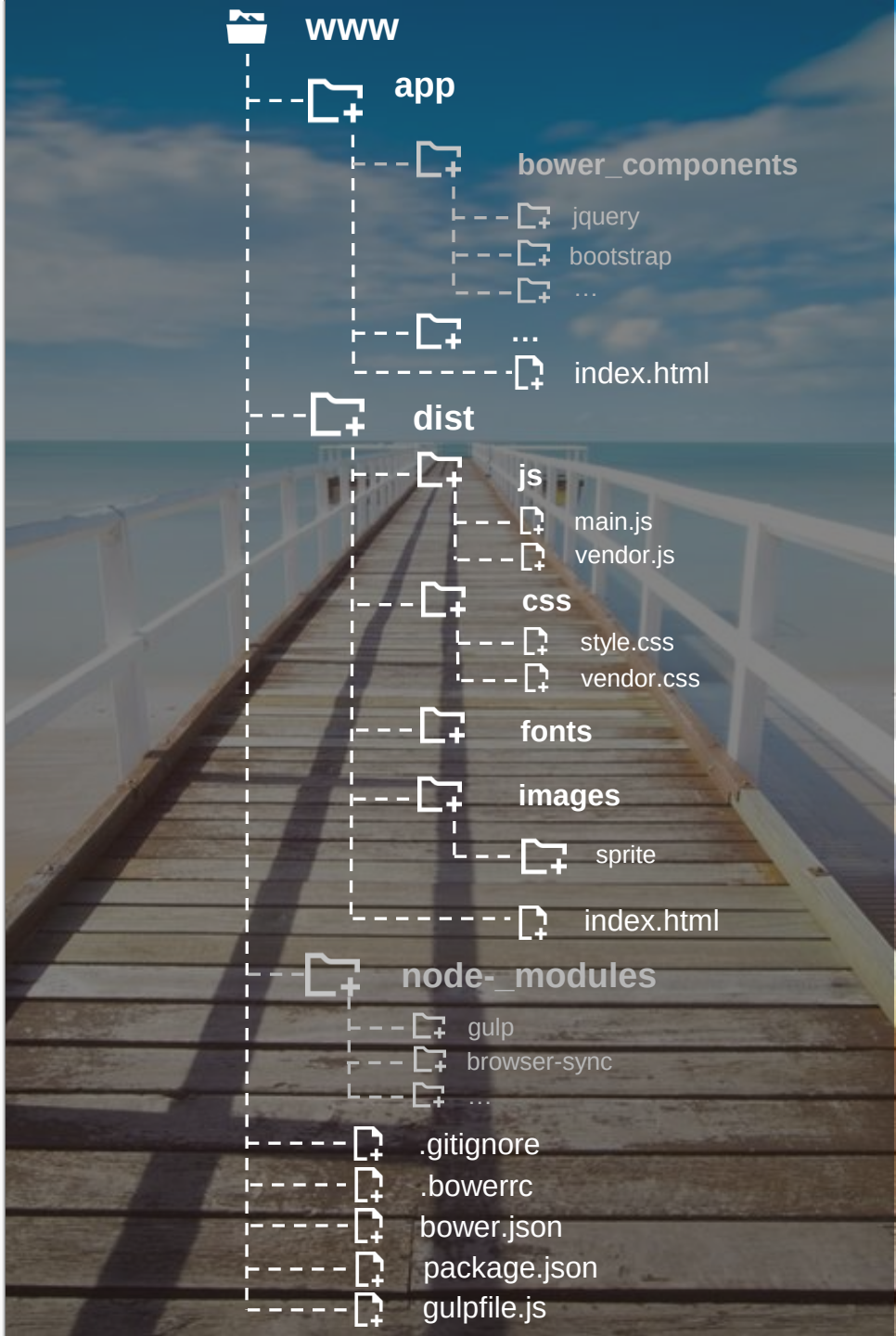
Установка:

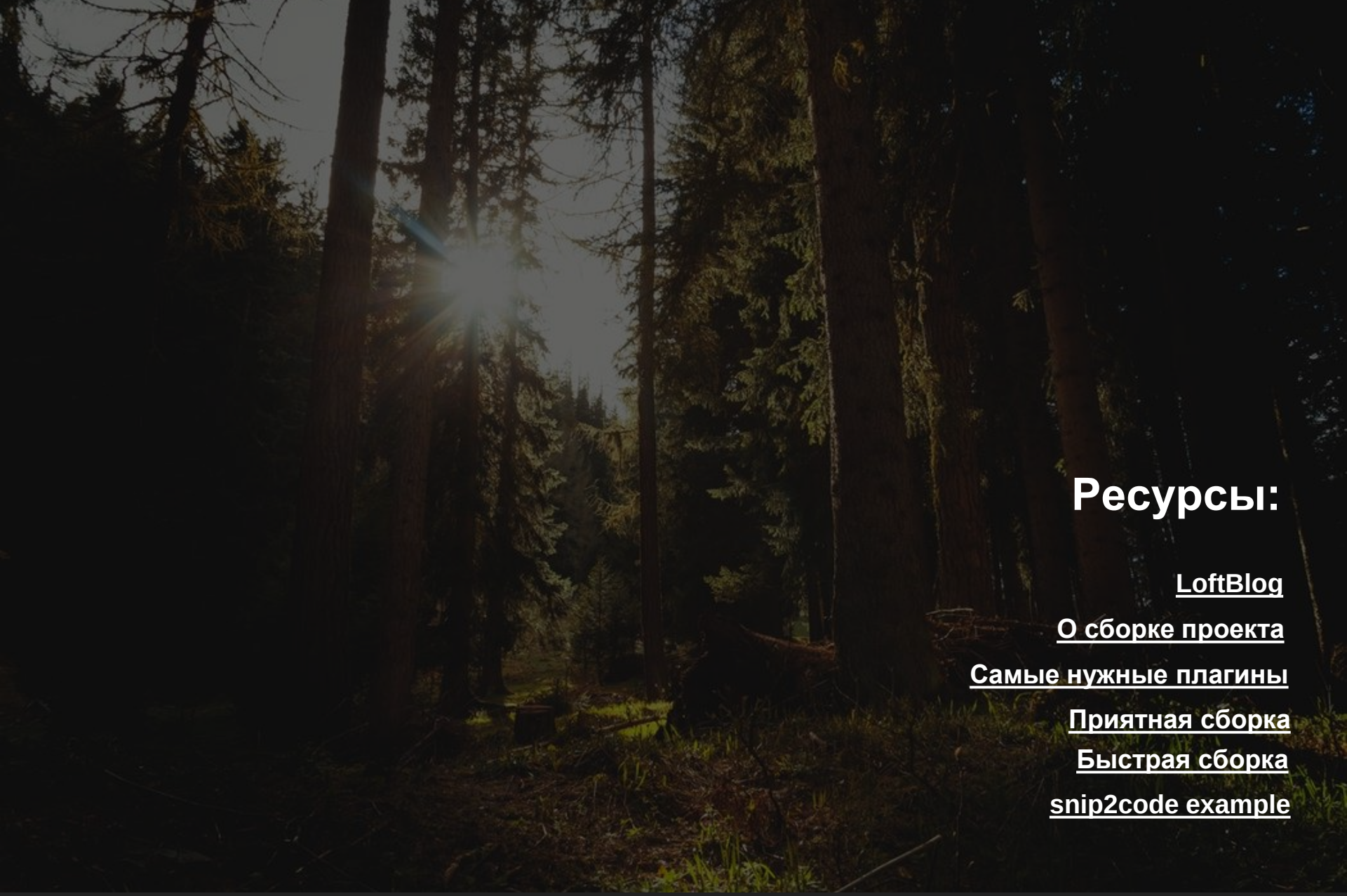
$ git clone https://github.com/lime7/www.git

$ npm i

$ bower i

$ gulp





Ресурсы:

LoftBlog

О сборке проекта

Самые нужные плагины

Приятная сборка

Быстрая сборка

snip2code example