02

Livereloading e sincronização entre dispositivos

Quando estamos em nosso ambiente de desenvolvimento constantemente modificamos arquivos. Para que a página exibida em nosso navegador seja sincronizada com as novas alterações, precisamos recarregá-la. Se esta tarefa já é um tanto incômoda de ser executada em nosso navegador, imagine se sua página também fosse visualizada em múltiplos celulares ou tablets ao mesmo tempo. Você precisará atualizar cada um deles!

Para complicar ainda mais, se você precisa realizar um scroll em sua página ou clicar em algum botão, provavelmente quererá executar as mesmas ações em todos os dispositivos móveis ao seu redor. Já imaginou o tempo que você gastará com todo esse processo? Quantas vezes você modificou um arquivo e esqueceu de recarregar a página?

Será que isso precisa ser sempre assim? Não há alguma forma de automatizarmos todo esse processo? Sim há e eu lhe mostrarei como, mas uma coisa você já sabe: usaremos o Grunt para isso!

Livereloading e sincronização com Grunt

O Grunt possui o plugin grunt-browser-sync (https://www.npmjs.org/package/grunt-browser-sync) que realiza o livereloading (atualização automática da página) toda vez que algum arquivo de nosso projeto mudar. Você pode até definir quais arquivos dispararão esse processo, mas na maioria das vezes você quererá que sejam todos. Além do livereloading, ele ainda sincroniza ações feitas localmente com seu browser preferido em todos os dispositivos que apontarem para a mesma página.

Por exemplo, se você clicar num botão, cada dispositivo que estiver apontando para a mesma página também sofrerá a ação, a mesma coisa quando você realizar o scroll da página. Fantástico!

Não podia ser diferente: instalamos o plugin através do npm:

```
npm install grunt-browser-sync --save-dev
```

E como todo plugin do Grunt, você precisa registrá-lo em seu `Gruntfile.js`:

```
grunt.loadNpmTasks('grunt-browser-sync');
```

Configurando a task

O plugin do Grunt disponibiliza a task `browserSync`. Nela criaremos o target `public` com a propriedade `bsFiles`. É nessa propriedade que indicamos quais arquivos serão monitorados, em nosso caso, todos dentro da pasta `public`:

```
browserSync: {
      public: {
          bsFiles: {
            src : ['public/**/*']
          }
      }
}
```

Podemos rodar a task no terminal:

```
grunt browserSync
```

A primeira coisa que você reparará é que seu terminal fica travado, mas não se preocupe. Isso acontece porque o Grunt dispara um processo na porta 3001 para monitorar seus arquivos. Caso você precise executar outra tarefa do Grunt precisará abrir outro terminal.

Outro ponto é a mensagem exibida no terminal:

```
BS] Copy the following snippet into your website, just before the closing </body> tag

<script type='text/javascript'>//<![CDATA[
;document.write("<script defer src='//HOST:3000/socket.io/socket.io.js'><\/script><script defer
//]]></script>
```

Para que a mágica do livereload funcione você deve copiar e colar o script que ele exibe no terminal em sua página. É esse script que acessa o processo que roda em nosso terminal. Toda vez que um arquivo for modificado o script será notificado recarregando seu navegador. Tudo uma maravilha, mas Imagine adicionar esse script em zilhões de página e ainda se preocupar em removê-lo na versão de distribuição do seu projeto?

É por isso que no lugar de abrirmos o arquivo diretamente do sistema de arquivos, podemos tornar esse processo acessível pelo browser, como se fosse um pequeno servidor web. Fazendo isso, o server, antes de devolver a página solicitada, terá uma chance de injectar o script de livereload para nós, para depois então enviá-la para o navegador.

Para habilitar esse pequeno servidor basta adicionarmos o target **options**.

```
browserSync: {
      public: {
          bsFiles: {
            src : ['public/**/*']
          },
         options: {
            server: {
                baseDir: "public"
            }
         }
      }
}
```

Repare que em `options` indicamos através da propriedade `server` o **baseDir** da sua aplicação. Encare-o como a raíz do seu projeto quando a URL `http://localhost:3002/` for acessada.

Ainda falta um detalhe! Queremos que ele também recarregue as páginas quando nossos arquivos `.coffee` e `.less` forem compilados. Para isso, precisamos integrar o `browserSync` com nossa task `watch` adicionando a propriedade `watchTask : true` em `options`:

```
browserSync: {
      public: {
```

```
            bsFiles: {
              src : ['public/**/*']
            },
           options: {
              watchTask: true,
              server: {
                  baseDir: "public"
              }
           }
       }
}
```

Ainda será necessário rodar as duas tasks: primeiro `browserSync` e depois a `watch`. Podemos conseguir isso facilmente registrando uma nova task chamada `server`:

```
grunt.registerTask('server', ['browserSync', 'watch']);
```

E por fim, rodando a task no terminal:

```
grunt server
```

Seu navegador abrirá automaticamente já apontando para o IP local da sua máquina exibindo nossa página `index.html`. Experimente alterar o HTML e veja o resultado instantaneamente em seu navegador. Se puder, abra o endereço em seu celular e depois clique em seu navegador no botão que existe mais texto. Seu celular também receberá o click, sincronizando não apenas os arquivos, mas suas ações.

Por fim, nosso script final com tudo que fizemos até agora fica assim:

```
module.exports = function(grunt) {

   grunt.initConfig({
      /* Copia os arquivos para o diretório 'dist' */
      copy: {
         public: {
           expand: true,
           cwd: 'public',
           src: '**',
           dest: 'dist'
         }
      },

      clean: {
           dist: {
               src: 'dist'
           }
      },

      useminPrepare: {
        html: 'dist/**/*.html'
      },

      usemin: {
```

```
    html: 'dist/**/*.html'
  },

  imagemin: {
    public: {
      expand: true,
      cwd: 'dist/img',
      src: '**/*.{png,jpg,gif}',
      dest: 'dist/img'
    }
  },

  rev: {
    options: {
      encoding: 'utf8',
      algorithm: 'md5',
      length: 8
    },

    imagens: {
      src: ['dist/img/**/*.{png,jpg,gif}']
    },
    minificados: {
      src: ['dist/js/**/*.min.js', 'dist/css/**/*.min.css']
    }
  },

  coffee: {
    compilar: {
      expand: true,
      cwd: 'public/coffee',
      src: ['**/*.coffee'],
      dest: 'public/js',
      ext: '.js'
    }
  },

  less: {
    compilar: {
      expand: true,
      cwd: 'public/less',
      src: ['**/*.less'],
      dest: 'public/css',
      ext: '.css'
    }
  },

  watch: {
    coffee: {
      options: {
        event: ['added', 'changed']
      },
      files: 'public/coffee/**/*.coffee',
      tasks: 'coffee:compilar'
    },

    less: {
      options: {
```

```
          event: ['added', 'changed']
        },
        files: 'public/less/**/*.less',
        tasks: 'less:compilar'
      },

      js: {
        options: {
          event: ['changed']
        },
        files: 'public/js/**/*.js',
        tasks: 'jshint:js'
      }
    },

    jshint: {
      js: {
        src: ['public/js/**/*.js']
      }
    },

    browserSync: {
      public: {
        bsFiles: {
          src : ['public/**/*']
        },
        options: {
            watchTask: true,
            server: {
                baseDir: "public"
            }
        }
      }
    }

});

//registrando task para minificação

grunt.registerTask('dist', ['clean', 'copy']);

grunt.registerTask('minifica', ['useminPrepare',
                                'concat', 'uglify', 'cssmin', 'rev:imagens','rev:minificados',

grunt.registerTask('server', ["browserSync", "watch"]);

// registrando tasks
grunt.registerTask('default', ['dist', 'minifica', ]);

// carregando tasks
grunt.loadNpmTasks('grunt-contrib-copy');
grunt.loadNpmTasks('grunt-contrib-clean');
grunt.loadNpmTasks('grunt-contrib-concat');
grunt.loadNpmTasks('grunt-contrib-uglify');
grunt.loadNpmTasks('grunt-contrib-cssmin');
grunt.loadNpmTasks('grunt-usemin');
grunt.loadNpmTasks('grunt-contrib-imagemin');
grunt.loadNpmTasks('grunt-rev');
```

```
  grunt.loadNpmTasks('grunt-contrib-coffee');
  grunt.loadNpmTasks('grunt-contrib-less');
  grunt.loadNpmTasks('grunt-contrib-watch');
  grunt.loadNpmTasks('grunt-contrib-jshint');
  grunt.loadNpmTasks('grunt-browser-sync');
}
```