

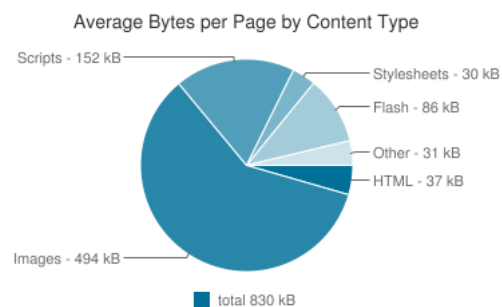
## Otimização de imagens

### O peso das imagens na web

Imagens são também fortes candidatas a otimizações. Quão importante será fazer isso?

O **HTTPArchive.org** armazena informações históricas coletadas mensalmente sobre os 17 mil sites mais acessados da Internet mundial. Com base nelas, compila alguns gráficos e dados interessantes.

E, com relação a imagens, os estudos mostram que mais de 60% do peso de uma página está nelas:



Se conseguir otimizar um pouco as imagens, o resultado final será de grande impacto para a página!

### A árdua tarefa de otimizar imagens

O problema é que sua aplicação pode ter um grande número de imagens ou talvez novas imagens sejam adicionadas periodicamente, o que demandará do designer ou do desenvolvedor que as otimize.

Outro ponto é que otimizações devem ser feitas apenas no projeto para distribuição, preservando as imagens originais que poderão ser retocadas sempre que necessário. Realizar esse processo manualmente pode ser algo impraticável.

Se conseguirmos automatizar essa tarefa, tanto o desenvolvedor quanto o designer poderão ter mais tempo para outras tarefas. É justamente o que aprenderemos a fazer.

### Otimizando imagens com Grunt

O Grunt também permite automatizar uma tarefa como essa através do **grunt-contrib-imagemin**. Sua instalação é como qualquer plugin do Grunt, logo, como qualquer outro módulo do Node.js:

```
npm install grunt-contrib-imagemin --save-dev
```

Como qualquer plugin do Grunt ele precisa ser carregado no `Gruntfile.js` :

```
grunt.loadNpmTasks('grunt-contrib-imagemin');
```

Por fim, resta apenas configurar a task também no `Gruntfile.js` :

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

A propriedade `src` indica que queremos todos os arquivos png, jpg e gif que estiverem em qualquer subpasta dentro de `dist/img`.

A task acima criará as imagens otimizadas na pasta de distribuição do projeto, sendo assim, sua task deve ser registrada após a cópia dos arquivos:

```
grunt.registerTask('minifica', ['useminPrepare', 'concat',  
                                'uglify', 'cssmin', 'usemin', 'imagemin']);
```

A versão final do nosso script fica:

```
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'
```

```
    }  
  }  
  
});  
  
//registrando task para minificação  
  
grunt.registerTask('dist', ['clean', 'copy']);  
  
grunt.registerTask('minifica', ['useminPrepare',  
                                'concat', 'uglify', 'cssmin', 'usemin', 'imagemin']);  
  
// 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');  
}
```

No terminal já podemos rodar nosso script:

```
grunt
```

Várias informações serão exibidas, a diferença é que agora temos um relatório do imagemin:

```
Running "imagemin:public" (imagemin) task  
? public/img/grunt-banner.png (saved 20.58 kB - 18%)  
Minified 1 image (saved 20.58 kB)
```

Conseguimos uma redução de 18% numa png. Dependendo do arquivo, o resultado poderia ser mais surpreendente. Qualquer nova imagem em nosso projeto será otimizada na distribuição.