Instituto Politécnico de Santarém
Escola Superior de Educação

Design de interfaces dinâmicos em sistemas de gestão de conteúdos

Relatório de Estágio para a obtenção do grau de Mestre na área de Educação e Comunicação Multimédia

Ricardo Jorge Martins Pinto

Orientador:
Prof. Especialista Nuno Bordalo Pacheco

2013
Maio

Instituto Politécnico de Santarém
2013

***Design* de interfaces dinâmicos em sistemas de gestão de conteúdos**

Ricardo Jorge Martins Pinto

Design de interfaces dinâmicos em sistemas de gestão de conteúdos

Relatório de Estágio para a obtenção do grau de Mestre na área de Educação e Comunicação Multimédia

Ricardo Jorge Martins Pinto

Orientador:

Prof. Especialista Nuno Bordalo Pacheco

2013

Maio

Índice

Índice 2
Lista de figuras 4
Lista de gráficos 5
Abstract 6
Keywords 6
Resumo 7
Palavras-chave 7
Definições e Acrónimos 8
Agradecimentos 9
Introdução 10
Parte 1. Enquadramento e tecnologias a utilizar no projeto 11
1.1 Contexto do projeto 11
1.2 Objectivos 11
1.3 O CMS 12
1.4 *Responsive Web Design* 13
Parte 2. Fases de construção do projeto 19
2.1 Processo de desenvolvimento com o *Wordpress* 19
2.1.1 Estrutura comum de temas do *Wordpress* 20
2.1.2 Filtros de interpretação dos ficheiros de *template* 22
2.1.3 Hierarquia de ficheiros no sistema de *templating* do *Wordpress* 22
2.1.4 Registo de scripts externos e internos em ficheiros de *templating* do *Wordpress* 23
2.1.5 *Permalinks* ou URLs permanentes do *Wordpress* 23
2.1.6 *Plugins* do *Wordpress* 25
2.2 Framework CSS e Javascript 27
2.2.1 Recursos de interface do utilizador nativos no *WordPress* 29
2.3 *Media Queries* 29
2.4 SVG e *Sprites* CSS 31
2.5 Fundamentos de ergonomia no Interface 32
2.6 Hierarquia do código e conteúdo 33
2.6 Base de construção 35
Parte 3. Metodologia e análise de dados 38
3.1 Testes informatizados empregues 38
3.1.1 Testes nativos com WP_DEBUG 38
Feedback da função WP_DEBUG: 38
3.1.2 *Theme Tester* 39
3.1.3 *Selenium IDE* 40

3.1.4 Avaliação W3C do *website* MECM em dispositivos móveis 41
3.2 Universo Alvo Versus Universo Inquirido 43
3.3 Objetivos da investigação proposta 43
3.4 Recolha de dados por questionário 43
3.5 Análise dos dados 43
3.5.1 Questões e respostas obtidas 44
Conclusão 53
Bibliografia 56
Anexos 58
Configuração de permissões e acesso a ficheiros e pastas no *Wordpress*. 59
Taxonomia Classificativa para o Design de Interfaces Dinâmicos em Sistemas de Gestão de Conteúdos 60
Vista comum de ficheiros de *template* interpretados pelo *Wordpress* 61
Performance do *Website* 62
Acesso à Internet em banda larga móvel 62
Questionário 63
Visualização do *website* de MECM em diferentes dispositivos 65

Lista de figuras

Figura 1 - *Viewport* 14
Figura 2 – Tipos de *media* CSS2.1 15
Figura 3 – Dimensões e orientações de tipos de *media* 16
Figura 4 – Exemplos de diferentes tipos de *media screen* em dispositivos *Apple* 17
Figura 5 – Orientação do *layout* de dispositivos com tipos de *media* comuns 17
Figura 6 - W3C Mobile Development 20
Figura 7 - Registo de ficheiro Javascript interno no template 23
Figura 8 - Opções de ligações permanentes 24
Figura 9 - *Permalinks* 25
Figura 10 – Acesso à API do painel de administração do *WordPress* 26
Figura 11 - shortcode do slideshow 26
Figura 12 - slideshow 27
Figura 13 - Vista de conteúdos numa grelha com 12 colunas em *desktop* e *smartphone* 28
Figura 14 – Icones gerados a partir de SVG+CSS3 31
Figura 15 – Lei de Fitt 32
Figura 16 – Menu de navegação 33
Figura 17 - Hierarquia do código 34
Figura 18 - Hierarquia de conteúdos 35
Figura 19 – Representação de acesso às funções do *Wordpress* 37

Lista de gráficos

Gráfico 1 - Questão 1..... 44
Gráfico 2 - Questão 2..... 45
Gráfico 3 - Questão 3..... 45
Gráfico 4 - Questão 4..... 46
Gráfico 5 - Questão 5..... 47
Gráfico 6 - Questão 6..... 48
Gráfico 7 - Questão 7..... 49
Gráfico 8 - Questão 8..... 50
Gráfico 9 - Questão 9..... 51
Gráfico 10 - Questão 10..... 52

Abstract

This project is included within the 2nd year course unit of Seminar of the Master Degree in Education and Communication Multimedia (MECM) of the School of Education, Polytechnic Institute of Santarém (ESE/IPS).

The main objective of this project is to conduct a study of the integration of Interface Design for Dynamic Content Management Systems (CMS) through the analysis of the various stages of implementation of a practical work, a WordPress website, available for viewing at www.mecm.eraizes.com. In particular to conceive PHP functions with the WordPress API in its *templating* structure. To implement CSS Media Queries elements in order to conceive a responsive design, resizable across the desktop and mobile devices such as Smartphones and Tablets PC. The main challenge of this report will be to convince the reader of the need for the implementation of the objectives proposed in a CMS and the ramifications of the case study in layout optimization design of user interface for different types of devices.

We hope this paper to be a contribution for the academic community in general, regarding future Web developments in standpoint of the user design experience.

Keywords

"CMS", "Interfaces Dinâmicos", "Media Queries", "User Interface Design", "Wordpress".

Resumo

Este relatório está inserido no âmbito da unidade curricular de Seminário do 2º ano do Mestrado em Educação e Comunicação Multimédia (MECM) da Escola Superior de Educação do Instituto Politécnico de Santarém (ESE/IPS).

O principal objetivo deste projeto é o de elaborar um estudo da integração do *Design* de Interfaces Dinâmicos em Sistemas de Gestão de Conteúdos (CMS) através da análise das várias fases de implementação do trabalho prático, um *website* em *WordPress*, disponível para consulta em www.mecm.eraizes.com. Em particular conceber *functions* em *PHP* com a API do *Wordpress* na sua estrutura de *templating* e implementar os elementos *CSS Media Queries* para a conceção de um *Design* dinâmico, adaptável ao *desktop* e a dispositivos móveis como *Smartphones* e *Tablets-PC*. O principal desafio deste relatório será o de convencer o leitor da necessidade da implementação dos objetivos propostos num CMS e das ramificações do estudo do caso na otimização de *layouts* no *design* de interface do utilizador em diferentes tipos de dispositivos.

Esperamos que este documento seja um contributo para a comunidade académica no geral em futuros desenvolvimentos *Web* do ponto de vista do *design* da experiência do utilizador.

Palavras-chave

"CMS", "Design de Interface do Utilizador", "Interfaces Dinâmicos", "*Media Queries*", '*Wordpress*".

Definições e Acrónimos

API - *Application Programming Interface* ou Interface de Programação de Aplicações, é um conjunto de atributos que permite aceder às funcionalidades de um sistema.

CMS – *Content Management System* ou Sistema de Gestão de Conteúdos

CSS – Cascading Style Sheets ou folhas de estilos. Conjunto de regras que determinam o aspecto de elementos *HTML*.

DOM – *Document Object Model* ou Modelo de Objeto de Documentos, especificação da W3C que define a estrutura de um documento *Web* independente da linguagem ou plataforma.

ESE – Escola Superior de Educação de Santarém

IPS – Instituto Politécnico de Santarém.

HTTP – *Hypertext Transfer Protocol* ou Protocolo de Transferência de Hipertexto. Protocolo utilizado na comunicação de páginas web que utiliza a linguagem *HTML*.

HTML – HyperText Markup Language. Linguagem utilizada para produzir páginas Web.

jQuery – Biblioteca de *Javascript* concebida para seleção e animação de elementos *DOM* na conceção de aplicações.

MySQL – SGBD que utiliza *Structured Query Language*

MECM – Mestrado em Educação e Comunicação Multimédia da ESE | IPS.

PHP – *Hypertext Preprocessor*. Linguagem dinâmica para processamento de dados do lado do servidor.

Responsive Web Design – RWD, Layout Web que se adapta a diferentes resoluções de ecrã.

SGBD – Sistema de Gestão de Bases de Dados.

SVG – *Scalable Vector Graphics*. Linguagem *XML* que descreve gráficos vectoriais bidimensionais de modo estático ou dinâmico.

Template – Conjunto de ficheiros de linguagem dinâmica (ex:*PHP*) com output em *HTML* aplicável a conteúdos multimodais que define o aspeto de uma página *Web* em contexto *CMS*.

Wordpress – CMS de código aberto desenvolvido em *PHP* e *MySQL*.

W3C – *World Wide Web Consortium*. Organização Internacional que define os standards da World Wide Web.

XML – *eXtensible Markup Language*, é uma recomendação da W3C, utilizada para descrever dados e partilha de informações entre outras linguagens**.**

XHTML – *eXtensible Hypertext Markup Language*, é uma variação do HTML que inclui regras de XML para melhorar a acessibilidade das páginas *Web*.

Agradecimentos

À minha esposa, Joana e filha Matilde, aos meus pais, José e Idalina, pelo apoio e paciência no decorrer desta etapa da minha vida académica. Ao orientador, o Professor Especialista Nuno Bordalo Pacheco pelas intervenções concisas na materialização do projeto de estágio. À Professora Doutora Maria Potes Barbas pelo acompanhamento e rigor científico na conceção do presente relatório. E por fim a todo o coletivo do CCTIC e departamento de *e-learning*, colegas Estudantes, Professores e Funcionários da ESES.

Introdução

Este relatório foi construído no âmbito da unidade curricular de Seminário do 2º ano do Mestrado em Educação e Comunicação Multimédia (MECM) da Escola Superior de Educação do Instituto Politécnico de Santarém (ESE/IPS).

O tema deste projeto surge da necessidade de uma visualização correta das páginas *Web* e/ou código HTML gerado pelos CMS em dispositivos móveis (*Smartphones, Tablet-PC's*). É nosso objetivo a implementação de técnicas RWD de forma transversal a todos os CMS, independentemente do seu modelo ou tipo, de comércio eletrónico, ensino à distância, gestão documental ou outro qualquer. A nossa área de intervenção será focada exclusivamente na estrutura de *templating* do CMS.

Com o advento das tecnologias e sistemas de informação a dimensão do interface gráfico a nível do utilizador tem vindo a assumir um papel dominante na aquisição de dados, com o objetivo primário de reduzir a distância entre Homem e máquina. No início da era digital a interação entre o utilizador e a aplicação ou *software* era restrita à linha de comandos, e consequentemente, restrita a um conjunto de utilizadores pré-definidos e habilitados a esse sistema. A evolução dos interfaces gráficos veio a ser determinante na democratização ao acesso da informação no contexto digital (Grudin, 2011). O utilizador já não necessita ter como pré-requisito, conhecimentos de programação para interagir com a aplicação. O incremento da literacia digital é consequência direta do Interface gráfico, e é através do estudo da usabilidade e acessibilidade dos sistemas que se afirma a implementação da ergonomia digital.

O *design* de sistemas interativos é um processo resultante do ciclo de vida de um software e das necessidades que este visa colmatar. A sua evolução é constante, e o seu formato altera-se de acordo com as nossas necessidades (Maragos, Potamianos, & Gros, 2008). Não há uma intenção explícita dessas necessidades, existem sim tendências de mercado, geradas por um produto virtual que concebe essas necessidades, se existe correio eletrónico, surge a necessidade de criar e gerir listas de endereços eletrónicos, para o envio de *newsletters* que promovem vendas de artigos, serviços ou a promoção de eventos e a partir daí seguem-se múltiplas aplicações, de código aberto, ou proprietárias, no browser ou no desktop. Temos assim um produto digital que integra as tendências do mercado e que responde às necessidades dos utilizadores.

A componente de psicologia no estudo da experiência do utilizador beneficia e habilita os profissionais do campo de estudo em causa com os elementos necessários para compreender os hábitos e necessidades dos utilizadores.

O estudo da experiência do utilizador foca os seus objetivos em simplificar processos na manipulação e aquisição de informações. Não é possível reeducar a disposição interna de cada individuo, contudo há mecanismos para contornar os traços da natureza desse indivíduo através da relação entre o *design* de um produto digital ou físico e a experiencia emocional que resulta da sua interação e manipulação (Christian & Beale, 2008, pp. 116-129). A repetição de ações elimina a complexidade dos sistemas, não só por si, mas em conjunto com o ambiente de desenvolvimento e claro o contexto.

Neste contexto em particular, o desenvolvimento de um produto multimédia, em concreto um *website* que irá servir como portal de interatividade e divulgação do Mestrado em Educação e Comunicação Multimédia.

Parte 1. Enquadramento e tecnologias a utilizar no projeto

1.1 Contexto do projeto

A singularidade do âmbito do projeto prático passa por informar a comunidade académica do percurso curricular do Mestrado de ECM ao garantir uma exposição adequada aos projetos desenvolvidos e a desenvolver no futuro. Pretendemos dotar o utilizador final com uma ferramenta de comunicação adequada ao seu perfil, uma plataforma de comunicação interativa aberta à discussão e a um discurso construtivo por parte dos utilizadores. Pretende-se reunir neste espaço digital a orientação e experiência dos docentes, também o percurso e conhecimento das necessidades do mercado de trabalho por parte dos ex-alunos do curso. Garantir aos atuais alunos um currículo atualizado do mestrado, e quem sabe um cenário para futuros alunos num espaço de interações rico e progressivo porque o contexto deste mestrado vai além do ambiente educativo e das tecnologias nele empregues, é sobretudo acerca da forma como comunicamos e concebemos uma matriz de conteúdos, de como identificamos as necessidades e comportamentos dos seus participantes.

A componente de estágio ocorre no departamento de *e-learning* da ESE. Este tema visa uma estratégia unificada para o desenvolvimento e *design* de interfaces dinâmicas em sistemas de gestão de conteúdos, através de um trabalho prático, análise e investigação da problemática num ambiente colaborativo. A conjunção do conhecimento de profissionais e especialistas no departamento foi determinante neste projeto, sobretudo ao estabelecer as prioridades técnicas e objetivos do presente relatório.

1.2 Objetivos

Desenvolver um interface dinâmico em CMS não significa recorrer a uma metodologia exata, isto porque não existe de momento qualquer API nativa a um gestor de conteúdos com essa funcionalidade. Assim sendo, recorremos a um conjunto de regras e boas práticas de programação e *web design* estabelecidas por convenções da W3C.

Enquanto Orientando, o estágio foi particularmente útil na construção da proposta para o trabalho prático, a transição entre o contexto empresarial e académico transporta uma trajetória do conhecimento adquirido. No contexto académico, das TIC o desenvolvimento é centrado nas necessidades do utilizador final, por outro lado as empresas solidificam a sua estratégia em audiências e públicos-alvo. O contraste entre a abordagem de cada uma dessas entidades sobre o individuo e o grupo, partilha de uma dificuldade comum, em identificar necessidades, quer seja de natureza material, ou de literacia informacional-digital, é neste ponto, neste núcleo de pesquisa, que divergem as intenções dos agentes educativos e empresariais. Contudo há uma simbiose a considerar,

a economia digital depende do nível de literacia digital dos seus consumidores, este é o papel fundamental do MECM, dotar os profissionais da área educativa, da comunicação e multimédia acerca do funcionamento e potencial das tecnologias empregues no curso. Formar quem utiliza e quem desenvolve as tecnologias de informação e comunicação.

O *design* centrado no utilizador é considerado como uma abordagem padrão em aplicações *web* e multimédia, o desafio é levar os utilizadores a participar nos conteúdos, esta proposta assenta na problemática e na estratégia a adotar na conceção de um interface dinâmico integrado num CMS no âmbito apresentado no ponto anterior. Até que a implementação deste projeto poderia ser centrada num contexto comercial, ao invés do âmbito proposto, isto porque o objetivo principal está em entregar a informação num universo multifacetado de dispositivos, navegadores *web* e sistemas operativos. Passamos a enumerar os objetivos:

1 A importância de um interface dinâmico em CMS;
2 O desenvolvimento de um interface dinâmico em CMS;
3 A implementação de um *layout* adaptativo;
4 A elaboração da experiência do utilizador em 3 cliques;

1.3 O CMS

Neste projeto iremo-nos focar num CMS em particular, o *Wordpress*, inicialmente desenvolvido enquanto plataforma de *blogging* em 2003. Esta ferramenta tem evoluído desde um sistema de publicação pessoal para um CMS, a sua arquitetura assenta em PHP e MySQL. Contudo e apesar da evolução das suas funcionalidades, a característica que mais distingue o *Worpress* é o facto de ser *user friendly*, aliás no âmbito do design de interface do utilizador, esta plataforma é uma referência, "*You can take advantage of its extensibility, friendliness in design, and function"* (Stern, Damstra, & Williams, 2010, p. 18). O seu formato respeita os *standards* da *Web* de acordo com a *W3C* e possibilita a expansão das suas funcionalidades, é um projeto de código aberto e encontra-se bem documentado.

Em primeiro lugar há que contextualizar o CMS, e o seu papel crescente nas instituições e empresas. Apontamos que o processo para a gestão de informação requer tempos de reação rápidos, esta observação parte do entrevistado, no artigo *Drupal Moves Into The White House* do *New York Times*, Ericsson, chefe-executivo da Acquia, uma *startup* de serviços web baseados em Drupal (Vance, 2009). Estruturalmente um CMS é um tipo de *software* que permite a publicação *online* e a gestão de conteúdos definidos por um utilizador de acordo com as suas funções e do seu nível de permissões nesse ambiente gráfico. Segundo *Kyle M. L. Jones*, e *Polly-Alida Farrington* (2011), "*Content Management is no longer optional"*, de acordo com os autores esta afirmação está diretamente relacionada com os princípios de interoperabilidade entre sistemas e uma gestão organizada dos conteúdos. Um exemplo dessa afirmação, comum a todos os CMS será a publicação RSS dos seus conteúdos por hierarquias ou temáticas e conetividade com leitores de *feeds online*, ou clientes de *email* como o *Outlook* ou *Thunderbird*. Essa inferência hierárquica dos conteúdos reduz a complexidade de um problema, e

proporcionar ao utilizador essa solução resolve a dificuldade no acesso à informação. Por outro lado a função e objetivos de um interface gráfico reside na perceção do conteúdo, navegação estruturada e consistente, princípios fundamentais na interação Homem-máquina, no *design* e experiência do utilizador. O que nos transporta ao objetivo de conceber a experiência do utilizador em três cliques num ambiente dinâmico. O termo multimédia preconiza o desenvolvimento de experiências interativas, e em síntese, interfaces dinâmicas. Ir além das boas práticas de usabilidade e acessibilidade, assim motivar o utilizador é a convenção deste trabalho prático. Contudo há uma dinâmica em progresso, de prioridades partilhadas que servem o lado do cliente, do utilizador final: O interface da aplicação ou *website* pode beneficiar de tecnologias emergentes como o HTML5, e *Cloud Computing*, contudo a evolução dos sistemas de informação compromete o futuro da aprendizagem de quem desenvolve e das empresas e instituições que investem em expectativa, como identificar o essencial do não essencial?

Esta é a promessa dos sistemas de código aberto e proprietários e serviços de *Cloud Computing*, nas suas instâncias do lado do servidor, como o *Cpanel*, *Parallels*, *Azure*, *Amazon Web Services*, e a sua famosa instalação de três passos do *Wordpress*, esta acessibilidade é o que permite verificar a validade deste CMS no contexto institucional e empresarial. Dos principais CMS, o *Wordpress* destaca-se pela sua rapidez na interpretação ou *query* SGBD, o que se deve à sua estrutura NoSQL (Not Only SQL), ou seja não encarrega os conteúdos do *website* exclusivamente à base de dados, esta informação é repartida também pela linguagem de servidor PHP e pela linguagem de *scripting* Javascript, esta arquitetura do sistema traduz-se num ambiente leve e rápido em comparação outros CMS. Há medida que a informação é introduzida na base de dados esta cresce até gerar situações criticas como erros de memória PHP, uma situação comum em *Joomla*, *Drupal* e *Moodle*, cuja única solução são a aquisição de alojamentos e servidores Web à medida para esses sistemas, mais dispendiosos e que não estão ao alcance de todos. Em contrapartida o *Wordpress* é o CMS que apresenta maiores problemas de segurança, que por delegar um maior conjunto de operações CRUD (Create, Read, Update e Delete) aos ficheiros PHP compromete e expõe em determinados momentos ficheiros de conteúdo sensível ao alcance de *malware* do tipo base64. É recomendado portanto que os ficheiros de uma instância *Wordpress* apresentem uma permissão de acordo com o anexo – Configuração de permissões e acesso a ficheiros e pastas no *Wordpress*.

1.4 *Responsive Web Design*

O termo *Responsive Web Design* foi definido por Ethan Marcotte[1] (2010) para *websites* que interpretam o *viewport* [2] (Connors, Adam; Google, 2010), um elemento de meta-dados que define a área de ecrã no dispositivo de destino. O *viewport* é inserido no elemento <*head*> de um documento HTML.

[1] http://alistapart.com/article/responsive-web-design
[2] http://www.w3.org/TR/mwabp/#bp-viewport

```
1  <!DOCTYPE html>
2  <html lang="pt">
3      <head>
4          <meta charset="utf-8" />
5
6
7          <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1" />
8
9          <title>Documento HTML</title>
10         <meta name="description" content="" />
11         <meta name="author" content="Ricardo" />
12
13         <meta name="viewport" content="width=device-width; initial-scale=1.0" />
14
15         <link rel="shortcut icon" href="/favicon.ico" />
16         <link rel="apple-touch-icon" href="/apple-touch-icon.png" />
17     </head>
```

Figura 1 - *Viewport*

Observemos a linha 13 da figura 1, da configuração do *viewport* no element <*head*> , em que o conteúdo do documento HTML deve ser apresentado na largura máxima pelo navegador e pelas dimensões máximas do dispositivo alvo. A 14 de dezembro de 2010 o artigo "*Mobile Web Application Best Practices*[3]" (Connors, Adam; Google, 2010) eleva o estatuto de configuração do *viewport* a recomendação oficial da W3C. Mais recentemente a 7 de maio de 2013[4] a organização responsável pelos *stantards* da *World Wide Web* publica o primeiro rascunho do artigo "*CSS Device Adaptation*" (Betts, Ryan - Adobe Systems; Lillesveen, Rune; Stenhaug, Øyvind - Opera Software; Rivoal, Florian, 2013), no qual apresenta as novas propriedades do *viewport*, em concreto o factor de zoom, escala e orientação do dispositivo-alvo. Voltando ao autor responsável pelo termo RWD, Marcotte refere que – "*We can design for an optimal viewing experience, but embed standards-based technologies into our designs to make them not only more flexible, but more adaptive to the media that renders them. In short, we need to practice responsive web design. But How?*"

Este autor acresce que se deve otimizar a experiência de visualização de conteúdos recorrendo a tecnologias *standard*, para que o design seja algo mais do que flexível, seja adaptativo aos dispositivos que a ele acedem. De imediato o autor passa a questão à entidade responsável pela padronização das tecnologias *web*, a W3C. E nos anos seguintes a organização tem respondido a esta questão com a introdução de novos elementos HTML e CSS para a visualização de documentos *web* em multiplataformas e nas mais variadas resoluções de ecrãs de *smartphones* e *tablet-PC's*. O artigo *Responsive Web Design* de Marcotte foi a origem de uma discussão construtiva na blogoesfera, contudo a sua intervenção veio levantar outras questões, "*we need to start admitting we don't know the exact conditions in which people view our websites*" (Coyer, Cannon, & Stewart, 2012, p. 6). Esta é a síntese de toda a problemática relacionada com RWD, quem produz e concebe produtos *web* e multimédia, tem de admitir desconhecer como é que o utilizador vai visualizar esses conteúdos, se através de um *smartphone*, a partir do desktop ou de

[3] http://www.w3.org/TR/mwabp/
[4] http://dev.w3.org/csswg/css-device-adapt/

qualquer outro dispositivo.

```
1  <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
2  "http://www.w3.org/TR/html4/strict.dtd">
3
4  <html xmlns="http://www.w3.org/1999/xhtml" lang="pt">
5      <head>
6          <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
7          <title>exemplo2</title>
8          <meta name="author" content="Ricardo" />
9
10         <link rel="stylesheet" type="text/css" href="core.css" media="screen" />
11         <link rel="stylesheet" type="text/css" href="print.css"media="print" />
12
13     </head>
14     <body>
15
16     </body>
17 </html>
```

Figura 2 – Tipos de *media* CSS2.1

Marcotte apresenta no seu artigo uma abordagem direta ao problema, de acordo com a figura 2 (linhas 10 e 11) se o CSS2.1 já prevê *media queries* para imprimir conteúdos de um *website* através do elemento *print* e o diferencia do tipo de media *screen* para os conteúdos disponibilizados no ecrã então nada mais natural que o CSS3 venha a englobar outros tipos de media, em que tipos de media refere-se a diferentes rácios de ecrã.

Em conjunto com o HTML5, o CSS3 disponibiliza a interoperabilidade necessária para reproduzir o conteúdo e design de forma adaptativa, em resumo o *viewport* no HTML acede às propriedades físicas do dispositivo e interpreta em conjunto com os *media queries* contidos no CSS a dimensão máxima adequada ao dispositivo. A 19 de Junho de 2012 surge a recomendação oficial da W3C para *media queries*[5] (Lie, Håkon; Rivoal, Florian; Van Kesteren, Anne - Opera Software; Çelik, Tantek - Stanford; Glazman, Daniel - Desruptive Innovations, 2012).

[5] http://www.w3.org/TR/css3-mediaqueries/

```
/*  #Mobile (Portrait)
================================================== */

    /* Nota: Design para uma largura de 320px */

    @media only screen and (max-width: 767px) {
        .container { width: 300px; }
        .columns, .column { margin: 0; }

        .container .one.column,
        .container .two.columns,
        .container .three.columns,
        .container .four.columns,
        .container .five.columns,
        .container .six.columns,
        .container .seven.columns,
        .container .eight.columns,
        .container .nine.columns,
        .container .ten.columns,
        .container .eleven.columns,
        .container .twelve.columns,
        .container .thirteen.columns,
        .container .fourteen.columns,
        .container .fifteen.columns,
        .container .sixteen.columns,
        .container .one-third.column,
        .container .two-thirds.column  { width: 300px; }
```

Figura 3 – Dimensões e orientações de tipos de *media*

Na figura 3 observamos a distribuição dos conteúdos por colunas semelhantes a uma tabela, mas ao contrário de uma tabela, essa distribuição de conteúdos não é fixa, ou seja ao se redimensionar o ecrã até um mínimo de 300 pixéis o conteúdo da página será sobreposto na vertical. Em primeiro lugar o dispositivo interpreta o *viewport* e em segundo interpreta o tipo de *media querie* apropriado às suas dimensões. Esta sequência automatizada na interpretação do código revela-se uma resposta eficaz à problemática expressa por Coyer. O responsável pelo desenvolvimento *web* desconhece os dispositivos de acesso aos conteúdos por parte dos utilizadores, contudo essa interpretação é agora feita de forma semi-automática, pelo dispositivo, e pelo browser, ao programador/designer cabe contemplar e atualizar os diferentes rácios de ecrã no documento CSS, contudo uma dimensão aproximada será o suficiente dado que o dispositivo seleciona o *media querie* com as dimensões apropriadas ao seu ecrã.

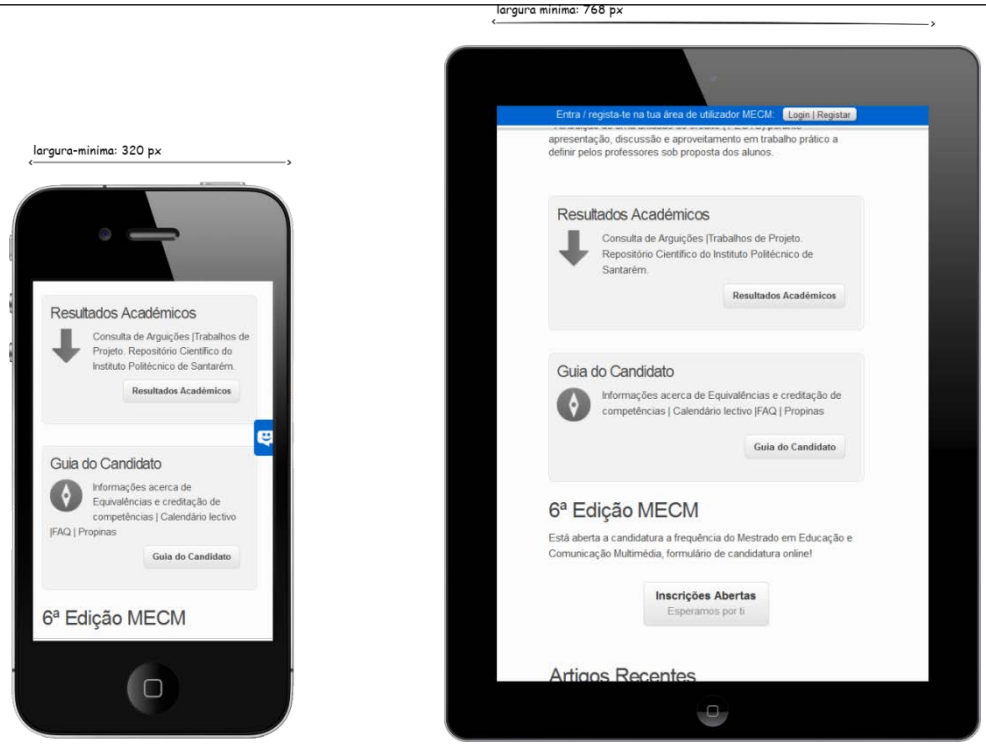


Figura 4 – Exemplos de diferentes tipos de *media screen* em dispositivos *Apple*

Observemos no exemplo da figura 4 um tipo de *media* de largura de 320 pixéis e um tipo de *media* de largura de 768 pixéis, um *iPhone4* iria interpretar o tipo de *media* de 320 pixéis porque este vai de encontro às suas dimensões, e o seu browser faria essa interpretação através do *viewport*, por outro lado o *iPad* iria interpretar o de 768 pixéis pela mesma razão. No entanto se o programador/designer não tivesse contemplado o tipo de *media* com a largura correspondente ou aproximada do *iPhone*, este iria interpretar a largura mínima no tipo de media disponível no documento que seria a do *iPad*. A necessidade de haver o mesmo elemento com uma largura mínima e uma largura máxima serve também a orientação do dispositivo, ou seja, a sua posição em paisagem ou retrato. Um fator importante para que o utilizador possa rodar o dispositivo de acordo com a sua preferência, e que o conteúdo não perca a sua formatação adaptativa. Contudo há que conceber que entre modelos do mesmo fabricante de *smartphones* e *tablets-PC*, essas medidas de áreas de ecrã variam, a figura seguinte representa uma abordagem às dimensões comuns entre dispositivos, haverá sempre um nível de aproximação às dimensões reais do dispositivo-alvo, e não um valor absoluto.

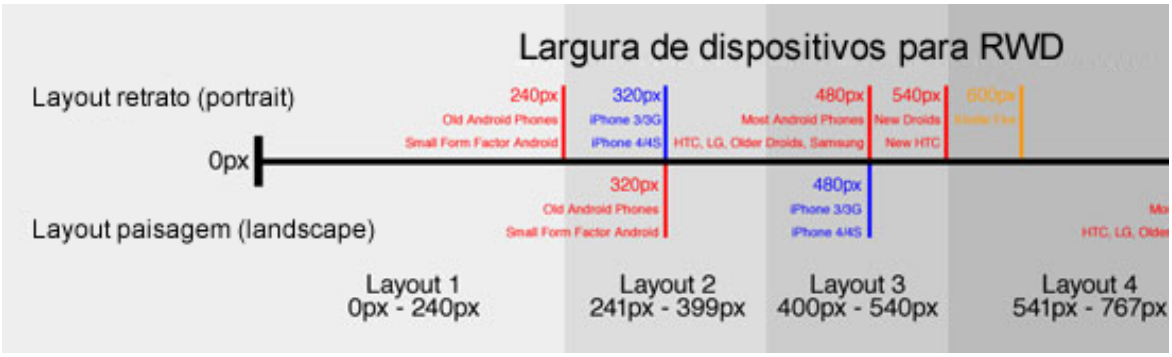


Figura 5 – Orientação do *layout* de dispositivos com tipos de *media* comuns

Prosseguimos a nossa investigação de acordo com o método abordado neste ponto para o desenvolvimento do nosso projeto. A existência de vários dispositivos com diferentes resoluções de ecrã justifica a necessidade de visualização dos conteúdos de acordo com as dimensões do dispositivo-alvo. No esquema apresentado na figura 5, podemos observar tipos de dispositivos comuns com as suas medidas em pixéis e os tipos de *media queries: media screen* sugeridos para desenvolvimento. A possibilidade de produzir num único documento CSS os *layouts* necessários para uma correta visualização do *website* em vários dispositivos face ao desenvolvimento de vários documentos e parâmetros para produzir layouts específicos a cada um desses dispositivos em separado previne a repetição de tarefas e acelera a produção do projeto. Não sendo necessário trabalhar com valores absolutos na produção dos *media screens* devido ao ajuste automático dos dispositivos, o utilizador beneficia de duas possibilidades para a visualização do website no smartphone ou tablet-pc, a visualização em layout paisagem ou retrato. Esta vantagem é acentuada nos dispositivos com dimensões de ecrã reduzidas, com larguras inferiores a 320 pixéis.

Passamos para a construção do nosso projeto com este método e tecnologias. Justificamos este enquadramento com a posição da W3C no artigo *Mobile Web Best Practices 1.0 – Basic Guidelines*[6] (Rabin & McCathieNevile, 2008), de que a informação disponível num *website* deve estar à disposição do utilizador independentemente do dispositivo deste, num limite razoável, ou seja não significa que essa informação tenha de estar representada da mesma forma em diferentes dispositivos. Algo a considerar, em parte devido à problemática na representação das classes CSS a conceber para a navegação do *website* em diferentes dispositivos, em concreto a hierarquia das classes.

[6] http://www.w3.org/TR/mobile-bp/#OneWeb

Parte 2. Fases de construção do projeto

Apresentamos aqui os fundamentos relacionados com os objetivos declarados anteriormente. Nesta parte do documento a nossa atenção foca-se nas tecnologias estruturais do projeto prático, do CMS ao RWD. Uma breve avaliação destes recursos é essencial para melhor entendermos as suas vantagens e limitações de acordo com o trabalho proposto. A evolução do desenvolvimento do *website* de MECM encontra-se integrada de acordo com a taxonomia classificativa declarada no **Anexo – Taxonomia Classificativa para o Design de Interfaces Dinâmicos em Sistemas de Gestão de Conteúdos**. Este modelo de desenvolvimento é inspirado nos conteúdos bibliográficos e nas diversas abordagens à problemática do desenvolvimento e produção *Web*. Nesta parte do relatório apresentamos os métodos de programação a aplicar no projeto prático.

2.1 Processo de desenvolvimento com o *Wordpress*

Na fase de produção, o projeto será desenvolvido numa máquina local, em ambiente de desenvolvimento LAMP (Linux, Apache, MySql, PHP). Dos benefícios existentes nesta abordagem, destacamos os seguintes:

1 Ter um registo de tudo o que já mudou e quando mudou;

2 Reverter o projeto para versões anteriores;

3 Aplicar o *redesign* do *layout* sem as preocupações inerentes de o fazer em ambiente de produção;

4 Controlo absoluto sobre o servidor, extensões PHP, Base de Dados e serviços apache;

5 Acesso direto a pastas e ficheiros para alteração de permissões;

Para os primeiros pontos enumerados utilizaremos um *software* específico para o controle de versões – GIT. Inicialmente desenvolvido por Linus Torvalds como ferramenta de gestão para coordenar o desenvolvimento do *kernel* do *Linux* (Spinellis, 2012).

Com o *Wordpress*, em termos de desenvolvimento, obtemos as seguintes funcionalidades base:

- Uma estrutura de base de dados;
- Utilitários de gestão da base de dados;
- Um sistema de *email*;
- Registo de utilizadores;
- Gestão de privilégios de utilizadores;
- É extensível (através de *plugins*);
- APIs Múltiplas.

A documentação necessária para utilização destas funcionalidades base do *Wordpress* encontra-se disponível no seu repositório oficial de documentação, o *Codex*.

Em primeiro lugar, há que definir a escolha no tipo de documento HTML do tema que pretendemos desenvolver. Isto é importante para a implementação dos tipos de media

CSS, que interpretam o dispositivo alvo, através da comunicação entre o software, o navegador, e hardware, os sensores de ecrã e orientação. Optamos então pelo HTML5, em concreto pelos elementos HTML contidos nesta ultima versão, que permite o acesso ao *hardware* dos dispositivos móveis, como sensores, funções de marcação, microfone e câmara fotográfica, entre outros, de acordo com o esquema apresentado na figura 6.

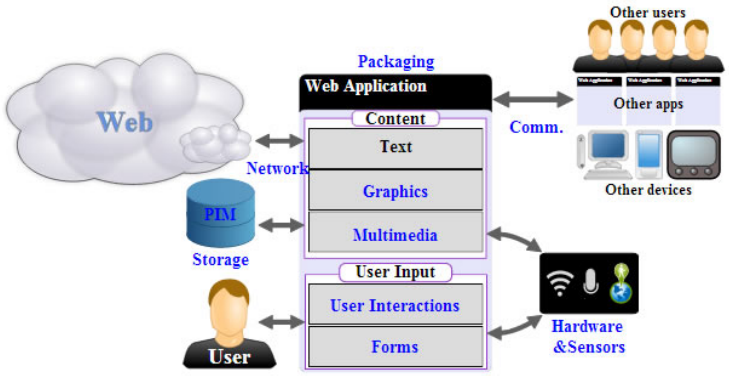


Figura 6 - W3C Mobile Development

É importante acrescentar a esta figura a declaração oficial do W3C sobre o HTML5, "*HTML5 is the cornerstone of the Open Web Platform, a full programming environment for cross-platform applications with access to device capabilities; video and animations; graphics; style, typography, and other tools for digital publishing; extensive network capabilities; and more*."

Há que observar contudo um potencial conflito entre as convenções W3C e a documentação oficial do repositório *Codex*[7] do *Wordpress* em que declara que todo o código HTML deve ser validado com o Validador W3C[8]. Contudo o código válido não é necessariamente uma prática correta, como é o caso do elemento <iframe>, embora haja uma correlação definitiva, depende da especificidade do projeto em questão. O *WordPress* é baseado na especificação do XHTML 1.0, que é a recomendação oficial do W3C desde 2000. Se observarmos o tema pré-definido na versão 3.5.1 do *Wordpress*, o *Twenty Twelve*, este apresenta o tipo de documento ou *Doctype* HTML5. Parece haver aqui um conflito, mas na realidade não há nenhum. HTML5 inclui uma secção de XML de serialização, o que na prática significa que XHTML5 pode ser utilizado. Ainda mais recentemente a 17 de Dezembro de 2012 surge a publicação: *HTML5 Definition Complete, W3C Moves to Interoperability Testing and Performance* (Berjon, et al., 2012), na qual a W3C declara, ainda que o *doctype* HTML5 não seja uma recomendação oficial, a sua especificação encontra-se finalizada para implementação por parte de empresas, instituições e ou programadores. Este documento e a aprovação do mesmo por parte dos parceiros da W3C, dota o nosso projeto com a legitimidade necessária para avançar com o tipo de documento HTML5.

A documentação oficial do *Wordpress* é extensa, não é o nosso objetivo um estudo do CMS, iremo-nos concentrar nas áreas e elementos necessários ao projeto a desenvolver.

2.1.1 Estrutura comum de temas do *Wordpress*

Salvo algumas particularidades do CMS, este será o palco primário de intervenção do nosso projeto, passamos a analisar a estrutura comum de um tema *Wordpress*:

[7] http://codex.wordpress.org/Validating_a_Website
[8] http://validator.w3.org/

- **Archive.**php – Apresenta as categorias por data, dia, mês ou ano;
- **Author.**php – Apresenta os artigos por autor;
- **Category.**php – Vista de um conjunto de artigos, por norma representada por títulos e um resumo dos artigos;
- **Footer.**php – Para a área de rodapé do website, reservado ao branding e direitos de autor, poderá conter também widgets do Wordpress, *scripts* como o Google Analytics são registados aqui através da função wp_register_script(), a qual descreveremos posteriormente;
- **Functions.**php – Este ficheiro é reservado às funções, filtros e parâmetros incrementados ao Wordpress;
- **Index.**php – Caso algum dos ficheiros não esteja presente na estrutura este será interpretado pelos filtros do sistema ou do ficheiro Functions.php, contudo a sua presença e obrigatória;
- **Header.**php – Por norma este é o recetáculo dos meta-dados do elemento HTML <head> que contem a declaração do tipo de documento, especificação de linguagem, e strings de metadados com informação do *website*, é comum introduzir-se aqui também o elemento <body> com o menu de navegação principal.
- **Page.**php – representa o layout de página específico, a partir do qual surge as variantes Blog-Page.php para uma vista dos artigos em blogue, ou Home-Page.php para uma vista personalizada da página principal;
- **Search.**php – Para a interpretação e visualização da função nativa de pesquisa do sistema;
- **Single.**php – Para uma vista exclusiva de um artigo em determinada categoria;
- **Sidebar**.php -
- **Styles.**css – Folha de estilos principal do tema, por norma responsável pelo tipo de *media screen*.

Este conjunto de ficheiros são denominados de *templates* ou modelos, em conjunto com a função e filtros descritas nos pontos seguintes são responsáveis por gerar as páginas do *Wordpress*. Estes ficheiros encontram-se localizados na pasta do tema, cuja representação da directoria é /wordpress/wp-contents/themes/tema/. Aqui podemos encontrar a pasta *images* reservada às imagens do *layout* do *website*, imagens de fundo, texturas, ícones e setas de navegação. De salientar que os principais ficheiros para a implementação do design adaptativo são o **header.php** e o **styles.css**, o *header*, é a área comum para os elementos que identificam a versão HTML, idioma, entre outros, o elemento ou regra, o *viewport*, que acede às propriedades dos dispositivos móveis, responsável pela identificação da área de ecrã. E ainda o ficheiro styles.css que guarda as regras de estilo do *layout* do tema, entre elas os *media queries* que apresentam a página *Web* com as dimensões apropriadas ao dispositivo de acesso. Há ainda um terceiro passo na implementação do RWD no nosso tema, o sistema de grelha de CSS, incluído na nossa *framework*.

2.1.2 Filtros de interpretação dos ficheiros de *template*

Para uma correta apresentação e ordenação dos conteúdos, a estrutura de *templates* do *Wordpress* é filtrada através da função get_query_template (), estes filtros a que o sistema se refere são na verdade *strings* que facilitam a pesquisa dos documentos do tipo *template*. Por exemplo, "{$tipo}_template" onde $tipo é o nome do ficheiro contido na hierarquia do tema sem a extensão .php, observemos uma lista completa desses filtros:

- index_template
- 404_template
- archive_template
- author_template
- category_template
- tag_template
- taxonomy_template
- date_template
- home_template
- front_page_template
- page_template
- paged_template
- search_template
- single_template
- text_template, plain_template, text_plain_template (all mime types)
- attachment_template
- comments_popup

2.1.3 Hierarquia de ficheiros no sistema de *templating* do *Wordpress*

Os ficheiros listados no ponto 2.1.1 são cruciais para o código gerado pelo CMS, este interpreta as *strings* ou filtros para popular a camada HTML com o ficheiro necessário, o qual interpreta a base de dados pelos conteúdos correspondentes. Tomemos por exemplo o url http://mecm.eraizes.com, em que um utilizador pesquisa por determinada categoria de artigos, no caso da categoria de Comunicação Interpessoal, o sistema encarrega-se de interpretar o ficheiro *archive.php* para resolver a função de pesquisa, este por sua vez interpreta a base de dados pelos artigos correspondentes à categoria de pesquisa e carrega os resultados na camada HTML, ou seja o código HTML gerado pelo *Wordpress* que responde o pedido do utilizador através do url http://mecm.eraizes.com/artigos/comunicacao-interpessoal/. Consultar o diagrama no **Anexo - vista comum de ficheiros de template interpretados pelo Wordpress** para uma leitura dos ficheiros de *template* comuns que são interpretados para gerar a camada ou documento HTML.

2.1.4 Registo de scripts externos e internos em ficheiros de *templating* do *Wordpress*

Uma função nativa do *Wordpress* fundamental para o seu sucesso, trata-se da possibilidade de expansão das suas funcionalidades através da função wp_register_script(). Em conjunto com os parâmetros necessários ($handle, $src, $deps, $ver, $in_footer) este processo permite introduzir ficheiros externos de javascript como o Google Analytics para análise do tráfego por parte dos utilizadores, jQuery para carregar a biblioteca de interface do utilizador nativa do *Wordpress*. Mas também permite o registo de ficheiros internos, ou seja no lugar de serem carregados a partir de servidores externos, estes podem ser colocados em subpastas do tema. Este processo é também utilizado pelos *plugins* do *Wordpress*, contudo consideremos esse atributo numa fase seguinte do projeto.

```
1  <?php
2  function meu_script_fantastico() {
3      wp_enqueue_script(
4          'script-fantastico',
5          get_template_directory_uri() . '/js/script_fantastico.js',
6          array( 'jquery' )
7      );
8  }
9
10 add_action( 'wp_enqueue_scripts', 'meu_script_fantastico' );
11 ?>
```

Figura 7 - Registo de ficheiro Javascript interno no template

No exemplo da figura anterior observamos o registo de um script interno num ficheiro de template, na linha 5 pudemos identificar a localização do ficheiro na pasta ‘js’, com o prefixo de get_template_directory_uri(), esta é uma hiperligação permanente, a sua função é determinante neste tipo de operações, dado que resolve a localização de ficheiros de diversos tipos num tema para instalação e ou importação e migração, que é a situação do nosso projeto.

2.1.5 *Permalinks* ou URLs permanentes do *Wordpress*

Permalinks são as URLs permanentes dos artigos, categorias, páginas, imagens, ou outros conteúdos e ficheiros no *Wordpress*. A hiperligação para cada conteúdo ou ficheiro deve ser permanente, e nunca mudam - daí o termo *permalink*. Existem três tipos básicos de permalinks no WordPress:

- mod_rewrite
- Structure Tags
- PATHINFO

O tipo *mod_rewrite* refere-se ao ficheiro *.htacess*, apresenta entre várias possibilidades, a alteração da ligação de acesso ao painel de administração do *WordPress*, contudo não iremos abordar este tipo de *permalink* por ser irrelevante para o tema deste relatório. O tipo *Structure Tags* é responsável pela gestão das ligações dos artigos e páginas no painel de administração do *WordPress*. É relevante na estrutura de navegação do *website*, representa um controle absoluto e simplificado na apresentação dos conteúdos ao utilizador, em termos de partilha e consulta desses mesmos conteúdos, observemos a

figura 8.

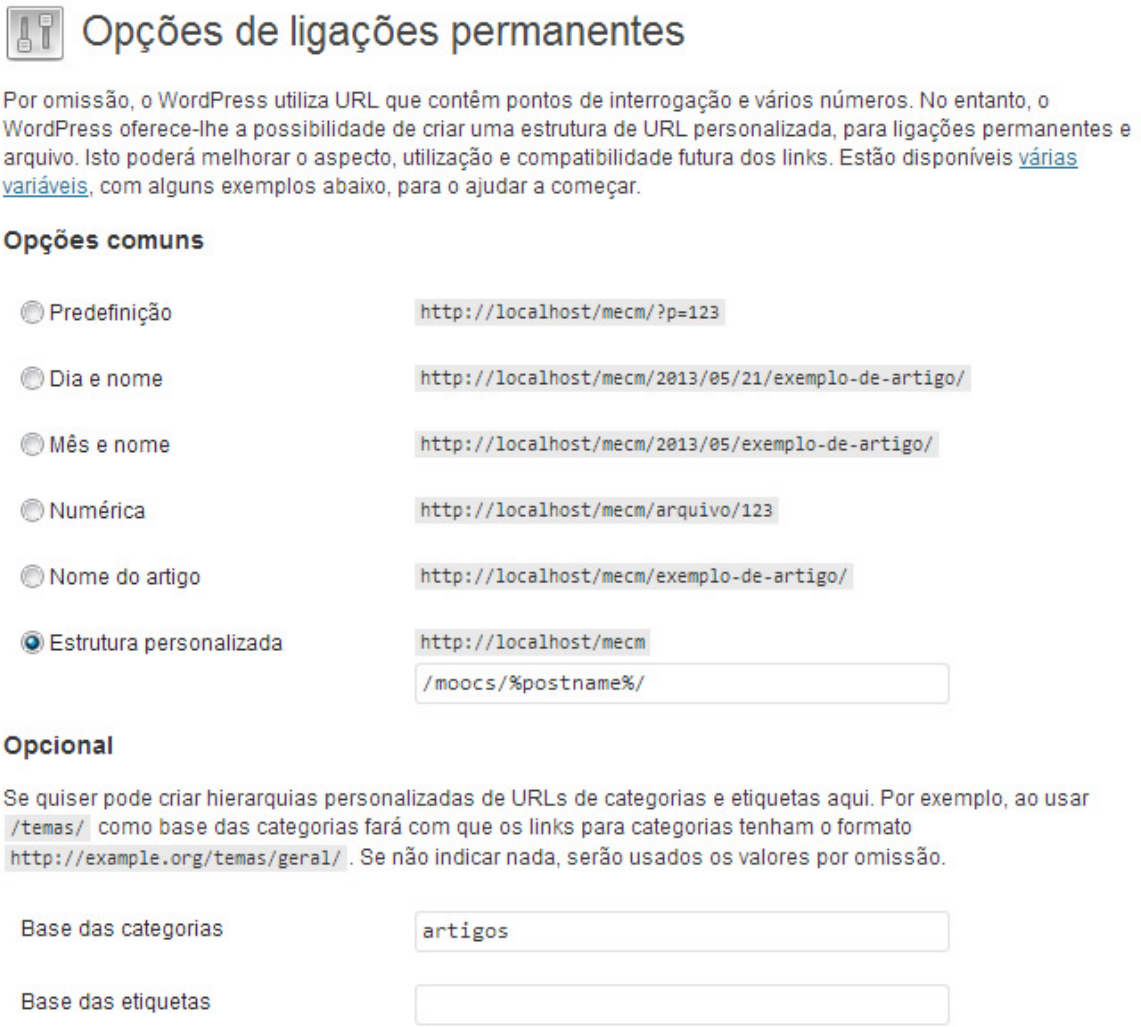


Figura 8 - Opções de ligações permanentes

Além das opções comuns, há ainda a possibilidade de uma estrutura personalizada através dos seguintes parâmetros:

- %postname% - Versão de nome simples do artigo.
- %post_id% - O número ID da página ou artigo.
- %category% - Versão da categoria.
- %tag% - Versão de nome simples da etiqueta do artigo.
- %author% - Versão de nome simples do nome autor.

Existem também parâmetros temporais, ano, mês, dia e hora. Ainda que não esteja diretamente relacionado com o nosso tema, esta referência a este tipo de *permalinks* é importante no contexto de acessibilidade do *website*, em concreto no mapa do website (Keith, 2005), e por isso parte do objeto de estudo. Contudo o tipo de *permalink* relevante para o *templating* do Wordpress é o PATHINFO. O objetivo dos *permalinks* deste tipo na estrutura de *templating* é o de atribuir aos recursos, como imagens ou ícones o caminho absoluto, de modo a que a estrutura do *WordPress* localize esses atributos de forma eficaz. Digamos que pretendemos mover o tema desenvolvido neste projeto para outra instância ou instalação do *WordPress* noutro domínio como www.eses.pt, e que o output do HTML gerado pelo CMS relativo a um ícone é o seguinte:

```
<img src="http://mecm.eraizes.com/wp-content/themes/mecm/imagens/sprite.png">
```

O *Wordpress* através do *permalink* localiza o recurso e apresenta no HTML gerado:

<img src="http://www.eses.pt/wp-content/themes/mecm/imagens/sprite.png">

O *permalink* é representado pela função *"template_directory"* , precedida pelo caminho relativo do recurso conforme a figura 9.

```
19
20
21
22  <img src="<?php bloginfo('template_directory'); ?>/imagens/sprite.png">
23
24
```

Figura 9 - *Permalinks*

Esta forma de gestão dos recursos através de ligações dinâmicas ou permanentes é o elemento chave no *templating* do *WordPress*, o que permite o desenvolvimento de um tema em ambiente de testes numa máquina local e a sua transferência ou migração para outro servidor e domínio.

2.1.6 *Plugins* do *Wordpress*

O núcleo do *WordPress* é projetado para ser leve, para maximizar a flexibilidade e minimizar o output do código HTML gerado pelo sistema para rápida interpretação do navegador. *Plugins* oferecem funções personalizadas e recursos para que cada utilizador do *Wordpress* de acordo com as suas permissões possa personalizar o site de acordo com as suas necessidades específicas.

Plugins são ferramentas que têm como propósito alargar a funcionalidade base do *WordPress*. De acordo com o *Codex*, a definição de *plugin* : *"A WordPress Plugin is a program, or a set of one or more functions, written in the PHP scripting language, that adds a specific set of features or services to the WordPress weblog, which can be seamlessly integrated with the weblog using access points and methods provided by the WordPress Plugin Application Program Interface (API)*[9]*."* – Para o caso iremos documentar apenas os *plugins* desenvolvidos no âmbito do tema proposto neste relatório, ou seja, o conjunto de funções acrescidas ao *WordPress* com o objectivo de *design* dinâmico.

O *slideshow* é uma forma generalizada para a disponibilização de informação rápida numa *home page*, através da apresentação rotativa de destaques em texto, ligações e imagens. Com a API do *WordPress*, acedemos à estrutura do seu código base para a gestão do nosso *slideshow*.

[9] http://codex.wordpress.org/Writing_a_Plugin

```php
/*
 * Adiciona o as opções para gerir o slidesow no painel de administração do wordpress
 */
function admin_menu()
{
    $page = add_options_page( __( 'Slideshow', 'wp-mecm-slider-conteudos' ),
                              __( 'Slideshow', 'wp-mecm-slider-conteudos' ),
                              'administrator', 'wp-mecm-slider-conteudos',
                              array( $this, 'admin_interface' ) );
}

add_action('wp_dashboard_setup', 'admin_interface');
```

Figura 10 – Acesso à API do painel de administração do *WordPress*

Na figura 10, desenvolvemos uma interface para a gestão do *slideshow* no painel de administração do *WordPress*, através de uma *action* da API, 'wp_dashboard_setup' . A partir daqui todas as funções desenvolvidas no *slideshow* podem ser atualizadas no painel de administração do *WordPress*.

Para a integração do *slideshow* na *home page* desenvolvemos um *shortcode*. Um *shortcode* permite incluir a funcionalidade desenvolvida num *plugin* num termo ou palavra-chave que identifica essa função, para utilização num *widget*, artigo ou página. Por exemplo para incluirmos o nosso *slideshow* (plugin) na página, utilizamos a função da API add_shortcode na construção do *plugin* conforme o seguinte: add_shortcode('slideraizes', 'wp-mecm-slider-conteudos). Esta ação e inserida após o código presente na fig.8.

Figura 11 - shortcode do slideshow

Assim para posicionar o *slideshow* no local pretendido, basta chamar o *plugin* com o *shortcode*: [slideraizes] na vista de edição de página do *WordPress*, de acordo com a figura 11.

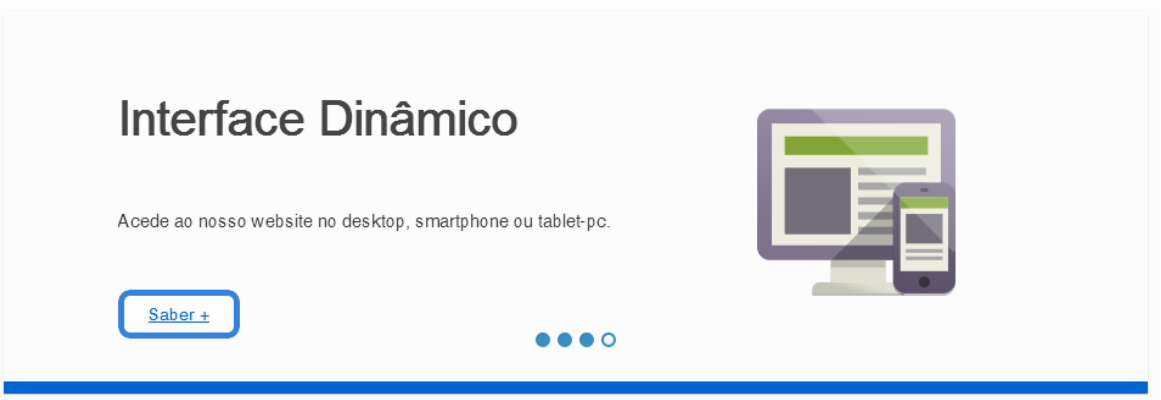


Figura 12 - slideshow

O resultado do *plugin* e *shortcode* na fig. 12, a vista do *slideshow* na página principal do *website*. Para a correta visualização do *slideshow* em diversos dispositivos, há também a introdução dos seguintes atributos para a classe *slideshow* do CSS:

```
#slideshow{
        width:800px;
        min-width: 320px;
        height:auto;
}
```

Com o atributo *width* definimos a largura máxima de 800 pixéis e com o atributo *min-width* definimos uma largura mínima de 320 pixéis, o *slideshow* acompanha assim a *grid* de colunas da *framework* para a visualização dos elementos nos vários rácios de ecrã entre o *smartphone* e o *desktop*. É importante a introdução do atributo *height:auto*, de modo a impedir que o *slideshow* fique esticado quando redimensionado, portanto a altura não pode ficar assente num valor fixo, mas sim num valor de ajuste automático, de acordo com a largura do ecrã.

A construção do *slideshow*, em CSS e PHP encontra-se documentada no anexo – Desenvolvimento do *plugin slideshow*.

2.2 Framework CSS e Javascript

Utilizar uma *framework* enquanto base do tema a construir para o *Wordpress*, acelera a fabricação dos conteúdos CSS, em pormenor, reduz a quantidade de elementos e regras a implementar, torna-se apenas necessário importar ou chamar essas classes de estilos em elementos dos *templates* e associar aos *ID's* das funções específicas do nosso tema. Aquilo que nos interessa em particular na utilização da *framework* é a sua estrutura de grelha, que divide a área de um documento em colunas. O conteúdo de um artigo *Wordpress* será apresentado de acordo com as dimensões do ecrã e de acordo com o

número de colunas disponíveis no CSS da *framework*.

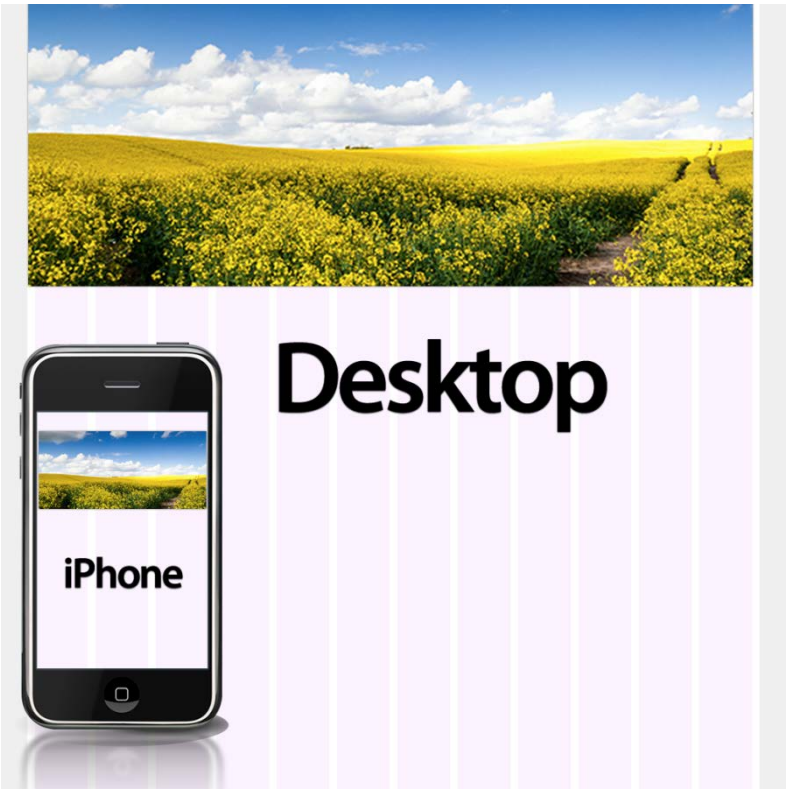


Figura 13 - Vista de conteúdos numa grelha com 12 colunas em *desktop* e *smartphone*

Conforme a figura 13, digamos que num monitor de 23 polegadas o utilizador terá a capacidade de visualizar um artigo com vários elementos multimédia em 12 colunas. E que o mesmo conteúdo acedido a partir de um *smartphone* com um ecrã de 3,5 polegadas será apresentado em 3 colunas. As colunas da *grid* definem a apresentação dos conteúdos de acordo com a dimensão do ecrã do dispositivo ou equipamento do utilizador. Está claro que para a grelha funcionar como esperado e imperativo a implementação do *viewport* no documento **header.php** do tema e a presença dos *media queries* nas regras de estilo.

A escolha sobre a *framework* utilizada no projeto recai sobretudo pela licença de utilizador, de domínio publico, conhecida como GPL. Entre as elegíveis de acordo com este critério encontram-se as seguintes:

- *Foundation* da *Zurb*[10]
- *Bootstrap* do *Twitter*[11]
- *Ink* do Sapo - Universidade de Aveiro[12]
- *Skeleton* de Dave Gamache[13]

Optamos pelo *Skeleton*, não só pelo seu ciclo de atualizações, mas também pelo seu suporte a versões anteriores do Internet Explorer. Encontra-se extensivamente documentada pelo seu autor, e ainda de salientar a sua crescente implementação em vários

[10] http://foundation.zurb.com/
[11] http://twitter.github.io/bootstrap/
[12] http://ink.sapo.pt/
[13] http://www.getskeleton.com/

projetos nos recentes anos, nos principais CMS, *Joomla*, *Drupal* e *Wordpress*.

2.2.1 Recursos de interface do utilizador nativos no *WordPress*

O *WordPress* oferece um conjunto de bibliotecas de *javascript* (*jQuery e jQuery UI*) para o interface do utilizador integradas no sistema, utilizar este recurso no desenvolvimento do tema WP e dos *plugins* necessários ao invés de registo de scripts externos. Em resumo, evitamos a redundância e aumentamos a performance do *Website* quando interpretado pelo navegador. Para integrar este recurso no desenvolvimento do *template*, basta acrescentar ao ficheiro functions.php a função: wp_register_script('jquery-ui'). Notamos no entanto que a performance do interface não apresenta melhorias significativas nos tempos apresentados nas auditorias do inspetor de elementos do navegador *Chrome*. A questão de performance é importante para a visualização do *website* em dispositivos móveis, as bibliotecas *jQuery* encontram-se localizadas no servidor de alojamento da nossa instância *WordPress*, por uma questão de eficiência, removemos o registo desses scripts nativos no ficheiro functions.php e registamos os scripts alojados pela *Google*:

```
wp_deregister_script('jquery-ui');

        wp_register_script('jquery-ui',
("//ajax.googleapis.com/ajax/libs/jquery/1.8.3/jquery.min.js"), false, '1.7.1');

wp_enqueue_script('jquery');
```

Após verificação observamos os tempos no gráfico gerado pelo inspetor de elementos do Chrome, uma melhoria significativa que passou de 2,6 ms para 0,9 ms o carregamento da biblioteca *jQuery* no *Website*, e um tempo total de 4 s, para o carregamento do *Website* no navegador. – Consultar Anexo Performance do Website.

2.3 *Media Queries*

Os *media queries* representam o núcleo para a implementação do *design* dinâmico no desenvolvimento *Web*. Encontram-se integrados na *framework Skeleton* no ficheiro styles.css, são o que permitirá o *website* de MECM chamar diferentes classes de estilos com base na largura do ecrã do dispositivo-alvo. Essa chamada é realizada através do *viewport*, explicado anteriormente, que inicia a consulta dos *media queries*, que posteriormente, chama os seus parâmetros.

Existem vários tipos de *media queries*, entre os principais destacamos os de impressão, voz, ecrã e *braille*. Apesar da utilidade de cada *media queries*, para a implementação do design dinâmico o nosso foco será no tipo de ecrã. Por definição, o valor do ecrã em pixéis. É representado da seguinte forma: “@media screen”. Para compreendermos melhor os seus parâmetros é necessário a vista do código que se segue:

```
@media screen and (max-width:1050px) {
   #page, .container, .main-header .container { max-width: 96% }
   #page { padding-top: 20px }
   .article { width: 66.3% }
   .sidebar.c-4-12 { width: 30% }
   .img, embed {
      max-width: 100%;
      height: auto;
   }
}
```

No exemplo acima é definido a largura máxima disponível do *website*, destacamos o atributo *max-width:1050px* – este terá como propósito a visualização do *website* no *desktop*. Os atributos das classes dependentes são apresentados de acordo com a largura máxima, em particular a classe *img, embed* que representa as imagens dos artigos do *website* e vídeos incorporados tem um atributo de largura máxima de 100%, dependente da largura máxima da página (96%), enquanto a altura da classe de imagem é relativa à largura.

Para uma adaptação a um *tablet-pc* implementamos o seguinte *@media screen* que apesar de herdar as classes e atributos anteriores define uma largura máxima de 728px e adapta a navegação do *website* de acordo com as dimensões de ecrã:

```
@media screen and (max-width:728px) {
   #header h1, #header h2 { margin: 10px 0 0 0 }
   .secondary-navigation {
      width: 100%;
      background-color: transparent;
   }
```

A implementação do tipo de *media querie* utilizado segue uma lógica de valores aproximados e não de valores absolutos, por exemplo:

```
@media screen and (max-width:400px) {
   article header { overflow: hidden }
   .main-header #s { width: 78% }
```

```
    #tabber ul.tabs li a { width: 70px }
    #commentform textarea { width: 90% }
}
@media screen and (max-width:300px) {
    nav fieldset, .js #navigation select {
        width: 100%;
        float: left;
    }
    .main-header #s { width: 70% }
    .related-posts li { width: 100% }
}
```

Neste *@media screen* acima representado, concebemos a adaptação do website entre os vários tipos de *smartphone* com largura máxima prevista entre os 300 e os 400 pixéis, independentemente do seu sistema operativo.

2.4 SVG e *Sprites* CSS

O SVG é um bom ajuste na representação de um ícone em contexto *web*. Por definição, um ícone é uma imagem ou símbolo para representar uma ação, a utilização do SVG na *web* e uma recomendação oficial da W3C. O SVG é um gráfico vetorial escalável, essa adaptabilidade é fundamental no tema proposto. Portanto é seguro admitir que esta abordagem serve os nossos propósitos. Como funciona o SVG no navegador? Este interpreta o SVG com base em parâmetros matemáticos, o que significa que o seu potencial de visualização num ecrã é exponencial, capaz de crescer ou diminuir de acordo com a resolução por pixéis do mesmo.

O suporte para SVG está disponível para a maior parte de sistemas operativos. Contudo em algumas versões anteriores do *Android* a tecnologia não é compatível, em todo o caso podemos disponibilizar no código uma imagem *bitmap* em *fallback*, ou seja após o pedido de acesso HTTP por parte do utilizador, é carregada a partir do servidor uma imagem do ícone num recurso de redundância.

Figura 14 – Icones gerados a partir de SVG+CSS3

Os ícones de partilha para redes sociais no *website* de MECM na figura 9 são gerados a partir de código SVG compatível com ecrãs retina, são imagens geradas por código, com dimensões adequadas aos dígitos, desta forma a interação por toque corresponde às normas de design de experiência de utilizador em interfaces *touch*.

As imagens são otimizadas através de técnicas de compressão, na verdade trata-se de uma única imagem em formato png para garantir transparência, o objetivo é o de apresentar diferentes partes dessa imagem em diferentes regiões do *website*, utilizando o CSS para a distribuição dessas partes através de coordenadas – esta técnica é denominada de *sprites*. Estas técnicas garantem rapidez no acesso e qualidade dos conteúdos, as funções e restantes scripts do *website* são carregadas em último, facilitando assim o acesso ao site através de ligações 3G, dado que as estatísticas mais recentes apontam para que cerca de 53% dos conteúdos online são acedido através de *smartphones* e *tablets pc's*.

2.5 Fundamentos de ergonomia no Interface

Presentemente o interface Homem-máquina vive um estágio de transição no *input* da informação, ao nível dos periféricos, da transição do terminal de texto para o ambiente gráfico, da evolução do próprio teclado, a introdução do rato, do aparecimento dos dispositivos móveis, até por ultimo à extinção do próprio periférico, com a introdução do *touch screen*, o interface táctil.

A afirmação chave da Lei de *Fitts* (Göktürk, 2008), é a de que o tempo necessário para mover o dispositivo apontador para um alvo é em função da distância do alvo e do seu tamanho. Noutros termos, quanto maior e próximo o alvo, mais rápido se consegue adquirir o alvo.

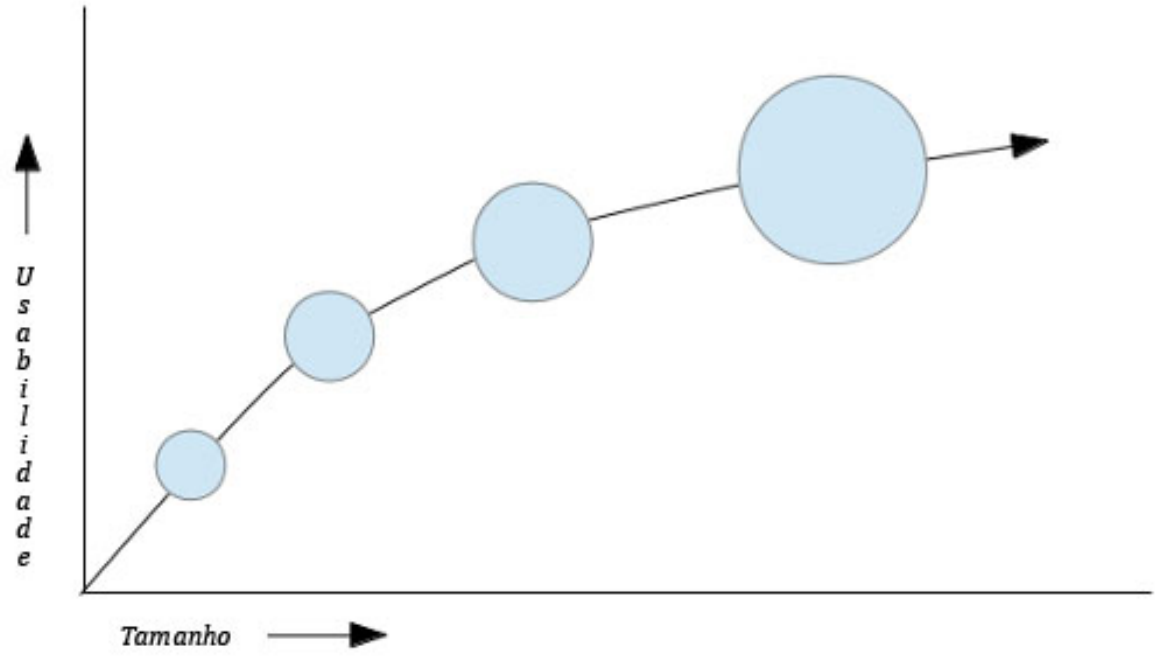


Figura 15 – Lei de Fitt

Assim, ao se desenhar um pequeno botão 10% maior, ele torna-se mais acessível, e desse modo diminui-se a distância e o tempo para o alvo, conforme representado na figura 9.

Segundo (Spitzer, 2012, p. 10), "*Mobile Design is mostly a process of evaluation and elimination: define the intended population, remove unneeded services, and strip away unnecessary data, as well as all extraneous visual and aesthetic elements*". Remover os dados desnecessários e definir a população-alvo é portanto uma prioridade no desenvolvimento de produtos orientados ao acesso móvel. Refinar e filtrar o acesso aos conteúdos de acordo com os dispositivos. E que tipo de conteúdos? A lógica seria, aqueles que não podem ser visualizados num ecrã menor ou que impedem o normal funcionamento do *website* nessas condições. No nosso caso, o elemento imediato será o menu de navegação, cujas dimensões são quebradas num ecrã pequeno, assim quando reduzido o menu de navegação converte-se em lista ordenada de acordo com a estrutura de menu fornecida pelo *Wordpress.*

Figura 16 – Menu de navegação

Introduzimos esta função no ficheiro **functions.php** sob a forma de uma condicional – se o ecrã do dispositivo for inferior a 800px de largura então o menu é representado em lista, conforme a figura 16. Esta regra aplica-se também aos conteúdos do tipo imagens e vídeos incorporados do *Youtube*, uma imagem original é redimensionada em várias e apresentada de acordo com o ecrã do dispositivo, enquanto a escala do vídeo é reduzida.

2.6 Hierarquia do código e conteúdo

O protocolo de acesso e comunicação à Internet, o HTTP, limita a visualização de conteúdos *Web*, na medida em que descarrega por cada pedido ou acesso apenas 2 ficheiros em simultâneo[14]. É por esta razão que muitos utilizadores esperam que o *website* carregue no seu navegador, e por vezes esse tempo de resposta desde o pedido de acesso até à resposta dos conteúdos desencoraja o utilizador de consultar o website em questão. Contornamos essa limitação através da gestão eficiente do código, através da

[14] http://www.w3.org/TR/html4/interact/scripts.html

ordem do carregamento de conteúdos conforme a fig.17.

```
<html>
<head>
    [css]<!--folhas de estilos no elemento head, carregam em primeiro-->
</head>
    <body>
        <h1>Titulo</h1>
        <div class="1 span8">1</div>
        <div class="2 span4">2</div>
        <div class="3 span8">3</div>
        <div class="4 span8">4</div>
        <div class="5 span4">5</div>
        <div class="6 span8">6</div>
        <div class="7 span8">7</div>
        <div class="8 span4">8</div>
    <footer>
        [script]<!--os scripts são carregados após todos os outros elementos -->
    </footer>

    </body>
</html>
```

Figura 17 - Hierarquia do código

Após o pedido de acesso, o *website* carrega o layout ou folha de estilos CSS em primeiro lugar no elemento <head>. As imagens são otimizadas através de técnicas de compressão e ícones gerados por código, estes conteúdos leves garantem rapidez no acesso e qualidade dos conteúdos, as funções e restantes scripts do *website* são carregadas em último no elemento <body> após o conteúdo (texto e imagens), facilitando assim o acesso ao site através de ligações 3G. Fundamentamos esta necessidade de hierarquia no código com os últimos dados estatísticos de Anacom que remontam a 2011. Em que se nota um acréscimo significativo no acesso à Internet de banda larga móvel, com uma taxa de aumento de 27,5% em relação a 2006 conforme o anexo - Acesso à Internet em banda larga móvel.

Em relação à hierarquia do conteúdo, como textos, vídeos e imagens, é importante definir nos ficheiros de *template* do tema *WordPress* desenvolvidos, uma relação entre a ordem dos elementos HTML e classes CSS utilizados. Isto para que o ajustamento da vista do *website* no desktop e a transição para a vista de *smartphone* seja apresentada de forma fluida e ordenada.

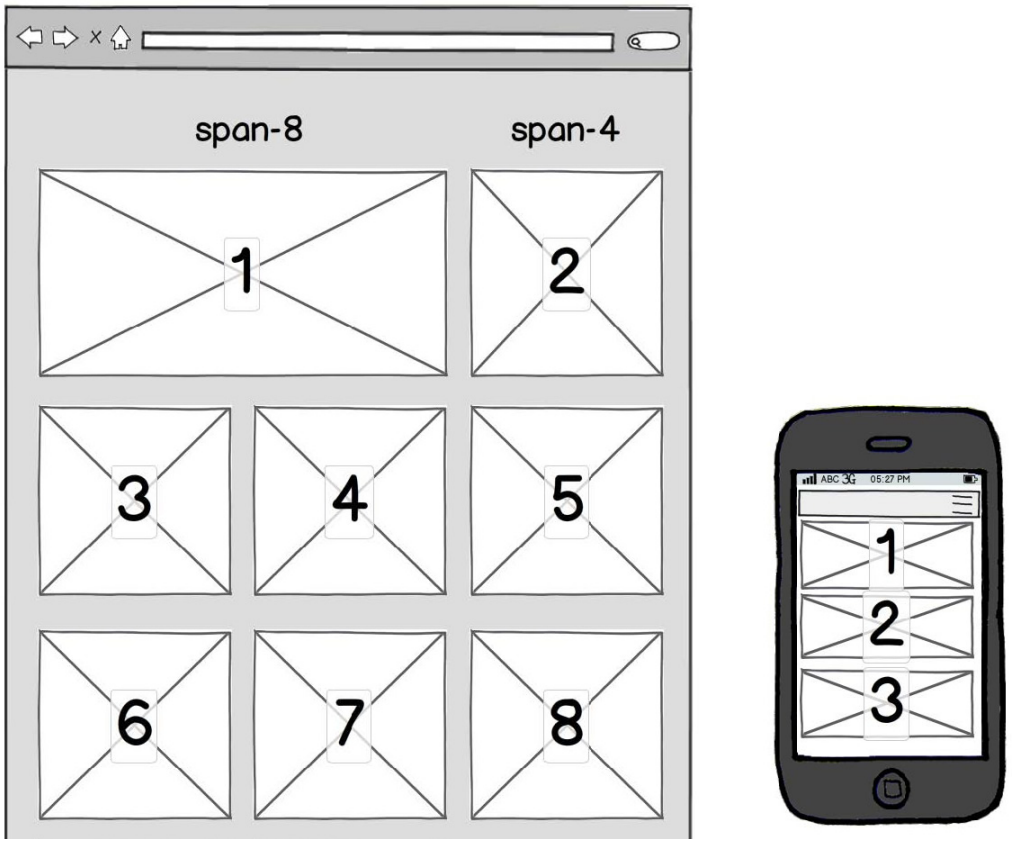


Figura 18 - Hierarquia de conteúdos

De acordo com a figura 18, os blocos de conteúdos num ecrã de menores dimensões encontram-se ordenados de acordo com as classes CSS utilizadas, a classe *span* representa a largura das colunas da *grid* contida na *framework Skeleton*, um total de 12 colunas nas quais os conteúdos são distribuídos. Assim o conteúdo mantém-se legível, porque respeita o tamanho da letra, sem a diminuir no caso de visualização dos conteúdos num ecrã menor. Remetemos o utilizador para o anexo - Visualização do *website* de MECM em diferentes dispositivos.

2.6 Base de construção

Elementos necessários à implementação do *design* dinâmico em Sistemas de Gestão de Conteúdos:

1. *Viewport*
2. *Media queries* do tipo @media screen
3. *Layout* CSS em *grid*

Onde colocar estes elementos:

Na estrutura de *templating* do CMS, em concreto nos ficheiros de estilos CSS, no ficheiro de index.php ou header.php, de acordo com o desenvolvimento do tema e do CMS. O ficheiro functions.php, é responsável pelo comportamento do tema, funciona como um *plugin* integrado no tema, com acesso às APIs do CMS, no *WorpPress*, em particular, através do ficheiro functions.php consegue-se aceder e alterar o comportamento do Painel

de Administração. Um exemplo dessas alterações na administração do CMS através do ficheiro functions.php, o rodapé ou *footer* que apresenta a mensagem: "*Thank you for creating with WordPress* ".

Com acesso à função footer_admin da API do dashboard WP, podemos alterar essa mensagem:

```
function remove_footer_admin () {
   echo "Mestrado em Educação e Comunicação Multimédia";
   }
   add_filter('admin_footer_text', 'remove_footer_admin');
```

O *WordPress* demarca-se de outros CMS através destas funcionalidades que controlam o comportamento não só do tema, mas também da estrutura do painel de administração. Embora esta situação não seja do âmbito do tema do nosso relatório, acreditamos que deve ser mencionada enquanto justificação do ponto de vista técnico na escolha do CMS para o nosso projeto. Numa perspetiva mais elaborada, desenvolvemos uma *widget* com um elemento HTML <iframe> para incorporar apresentações no painel de administração do *WordPress* visível aos utilizadores registados do Website, como sugestões de temas para desenvolvimento futuro. Acedemos novamente à API do painel de administração, em concreto à função de *widgets* - custom_dashboard_widgets:

```
function custom_dashboard_widgets(){
   wp_add_dashboard_widget('meecm_apresenta_tema_id', 'Apresentacao: Nice Reader', 'meecm_apresenta_tema');
   }

   function meecm_apresenta_tema() {
   echo '<p><iframe width="700" height="350" src="http://dev.ipsantarem.pt/perguntas/xindex.html" scrolling="no" ></iframe> </p>';
   }
```

Podemos determinar nesta fase que desenvolvemos com o *WordPress*, não desenvolvemos para *WordPress*. Ao acedermos ao núcleo das funções do CMS através de APIs na camada da estrutura de *templating* e a alterar o comportamento deste, estamos, portanto a desenvolver funcionalidades num ambiente colaborativo. Efetivamente estamos a produzir novas formas de apresentação e gestão de conteúdo com a sintaxe do *WordPress* em PHP (Schmidt & Etches, 2011).

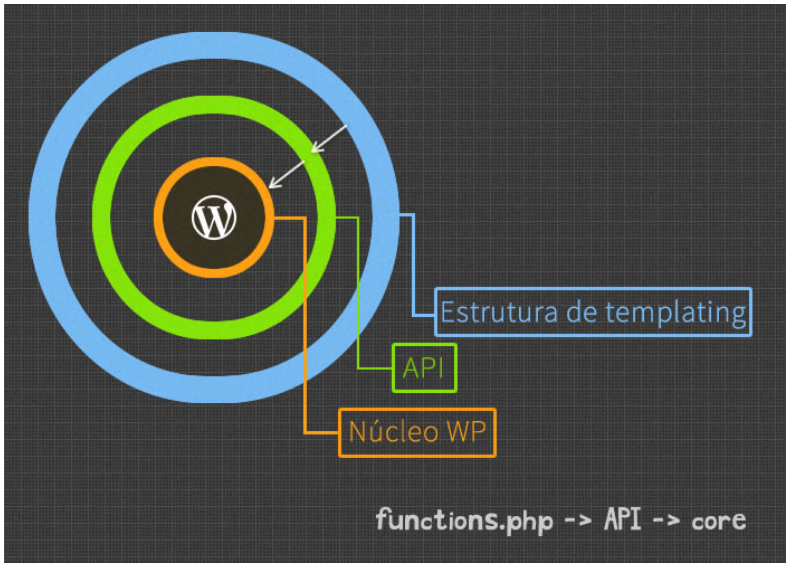


Figura 19 – Representação de acesso às funções do *Wordpress*

Na figura 19 observamos o método pelo qual acedemos ao núcleo do *Wordpress*. Com as referências das API's invocamos as funções do núcleo do CMS na estrutura de *templating*.

Estes são os diferentes tipos de API's disponíveis no *Wordpress*:

- **Plugin API** - Para implementar *Hooks*, *Actions*, e *Filters* no desenvolvimento e ou alterações a *plugins*.
- **Shortcode API** – Para implementar *shortcodes* das funções dos *plugins* ou funções nativas dos temas em artigos, páginas ou *widgets*.
- **Dashboard Widgets API** – Para desenvolver novos *widgets* no painel de administração.
- **Settings API** – Para adicionar novas definições ao painel de administração.
- **Widgets API** – Para o desenvolvimento de *widgets* no website.
- **Quicktags API** – Para acrescentar novos botões ao editor de texto HTML de artigos e páginas.
- **Rewrite API** – Para alterar ou rescrever os URLs.

Parte 3. Metodologia e análise de dados

A principal vantagem na escolha do tema proposto, consiste no facto de que um sistema de gestão de conteúdos apresenta-se como uma plataforma sujeita a testes de usabilidade e acessibilidade antes de se disponibilizada ao utilizador. Em particular, o *WordPress* é sujeito a testes unitários ao nível da sua linguagem de programação de base, o PHP, cada nova versão disponibilizada é sujeita a testes de *stress*, input de dados e navegação. As funcionalidades base oferecidas pelo *WordPress* são versões que passaram a fase de produção e de testes[15]. Desenvolvemos o nosso projeto, em concreto, a integração da *framework* escolhida numa plataforma estável e suportada com a documentação necessária para o desenvolvimento sustentado.

3.1 Testes informatizados empregues

Dirigimos portanto a metodologia empregue no curso desta investigação na utilização e manipulação da *framework* utilizada, transversal aos principais sistemas de gestão de conteúdos, *Drupal*, *Joomla* e *WordPress*. Essencialmente a composição do código nela presente, e das suas alterações faces às normas e recomendações da W3C. Conduzimos os testes ao *template* ou tema *WordPress* desenvolvido em contexto do *design* dinâmico apresentado neste relatório através de métodos nativos e externos. Em algumas situações específicas optamos por ignorar a informação gerada pelos testes, esta era derivada do nosso *workflow* como no caso do teste efetuado com o *plugin theme tester* para *WordPress*. Utilizamos o IDE (*Integrated Development Environment*) *Selenium* para testar as funcionalidades do *template*, que apresenta resultados específicos a nível de performance e acessibilidade, o que reduz significativamente a necessidade de um questionário, em particular reduz as questões a conceber nesse âmbito. Submetemos ainda o URL do *website* para a verificação da W3C em dispositivos móveis[16], o que contribui também para reduzir a especificidade das questões a formular aos inquiridos.

3.1.1 Testes nativos com WP_DEBUG

Um aspeto importante a considerar no desenvolvimento do projeto, em específico no *Wordpress*, é o de ativar no ficheiro **wp-config.php** a função WP_DEBUG, para isso basta alterar o seu parâmetro de *false* para *true*. Desta forma passamos a ter acesso à informação de *debugging* do tema e dos *plugins* instalados. Constatamos que através deste recurso WP_DEBUG, obtemos o feedback necessário do *Wordpress* para reparar o nosso código.

Feedback da função WP_DEBUG:

[15] http://core.trac.wordpress.org/browser/tests/branches/legacy
[16] http://validator.w3.org/mobile/

Notice: add_custom_image_header **não é usado** desde a versão 3.4! Utilize add_theme_support('custom-header', $args).

Este aviso refere-se à função da imagem de logotipo utilizada no tema, essa função permite ao utilizador alterar de forma automática com recurso a um campo personalizado o logotipo no *website*, constatamos que a versão atual do *Wordpress*, a 3.5 utiliza agora uma nova referência para essa função. De seguida alteramos o código no nosso ficheiro **functions.php** para incluir esta recomendação.

3.1.2 *Theme Tester*

Desenvolvido por *Simon Prosser* e por *Samuel Wood*, o *plugin Theme Tester*[17] disponível no repositório oficial do *Wordpress* faz exatamente aquilo que o seu nome indica, testa os temas desenvolvidos para Worpress. Emprega as orientações[18] definidas pelo *Wordpress* para o desenvolvimento de temas e compara com o código utilizado no tema para gerar um relatório dos erros encontrados.

Output do relatório do *theme tester* ao nosso projeto:

- RECOMMENDED: **add_custom_image_header** found in the file **functions.php**. Deprecated since version **3.4**. Use **add_theme_support('custom-header', $args)** instead.

Esta recomendação é uma redundância à informação gerada com o teste nativo anterior através da função WP_DEBUG.

- WARNING: **.gitignore .project** Hidden Files or Folders found

Definimos na parte 2.1 do presente relatório, em Processo de desenvolvimento com o *Wordpress* a utilização do Git para o controle de alterações do nosso projeto, este *software* deixa um rasto que foi captado pelo *Theme tester*. Devemos ignorar esta informação, não é relevante aos dados do tema, refere-se ao nosso *workflow*.

- RECOMMENDED: **get_bloginfo(wpurl)** was found in the file **functions.php**. Use **site_url()** instead.
- RECOMMENDED: **get_bloginfo(template_directory)** was found in the file **options.php**. Use **get_template_directory_uri()** instead.
- RECOMMENDED: **get_bloginfo(template_directory)** was found in the file **functions.php**. Use **get_template_directory_uri()** instead.
- RECOMMENDED: **get_bloginfo(stylesheet_directory)** was found in the file **functions.php**. Use **get_stylesheet_directory_uri()** instead.

Constatamos que as referências acima foram alteradas da versão 3.4 para a 3.5. Em simultâneo verificamos que o relatório deste teste com o *Theme Tester* é mais completo

[17] http://wordpress.org/plugins/theme-check/
[18] http://codex.wordpress.org/Theme_Review

que aquele obtido com a função nativa do *Wordpress*, WP_DEBUG.

3.1.3 *Selenium IDE*

O *Selenium IDE* providencia um ambiente de testes específico aos atributos de acessibilidade introduzidos no *website*. Disponibiliza um conjunto de parâmetros para testar as funcionalidades desenvolvidas no projecto.

O campo de pesquisa inserido no menu de navegação primário, é um campo de input de dados, a sua funcionalidade e testada e registada com o *Selenium*.

Teste de pesquisa		
type	id=s	faq
open	/	
clickAndWait	id=searchsubmit	
clickAndWait	css=a[title="Permalink to FAQ"]	

No teste de pesquisa, introduzimos o termo de pesquisa “FAQ” e este devolve a vista de pesquisa do *website através* do *permalink* com os resultados possíveis.

Teste de Login	
click	link=Login | Registar
clickAndWait	id=user-submit
Send	link=Início »

No teste de login verificamos o sistema de login introduzido no website, este regista/ inicia a sessão do utilizador no *website* e reencaminha-o para a página de início.

Teste de Navegação	
clickAndWait	link=Avaliação do Desempenho
clickAndWait	link=Publicações
clickAndWait	xpath=(//a[contains(text(),'Candidaturas')])[2]
clickAndWait	link=Artigos
clickAndWait	link=Mapa do Site
clickAndWait	css=li.page_item.page-item-191 > a
click	css=p.trigger.active
click	//div[@id='post-191']/div/div/div[12]/div/p
click	//div[@id='post-191']/div/div/div[4]/div/p

O teste de navegação revela as palavras chave correspondente aos conteúdos das

páginas contudo, falha no URL: http://mecm.eraizes.com/investigacao-edesenvolvimento/publicacoes/ , o elemento javascript responsável pelas expansão ou contração das publicações por ano, incompatível com as recomendações de W3C para o desenvolvimento Web em temos de acessibilidade, a recomendação oficial para este tipo de conteúdos é a criação de um índice. Contudo para a navegação de conteúdos extensos em dispositivos móveis consideramos que esta é uma abordagem pratica. A introdução de um índice tem como consequência o *scroll*, o que em conteúdos extensos não proporciona uma boa experiência de navegação ao utilizador em dispositivos móveis.

3.1.4 Avaliação W3C do *website* MECM em dispositivos móveis

A W3C disponibiliza uma ferramenta de avaliação *online* que verifica através do URL a validade do website em dispositivos móveis. Esta ferramenta analisa os atributos do código, imagens e restantes elementos, e fornece um relatório com a informação recomendada de acordo com os parâmetros da W3C para *websites* compatíveis com conteúdos para dispositivos móveis. Após a submissão do URL do nosso *website* obtivemos o seguinte relatório:

40%

FAILURES PER SEVERITY

- CRITICAL 1
- SEVERE 2
- MEDIUM 2
- LOW 4

FAILURES PER CATEGORY

- Rely on Web standards3
- Stay away from known hazards2
- Check graphics and colors2
- Keep it small2
- Use the network sparingly2

CRITICAL 1= The document contains a frame, frameset or iframe element

Why?

Most mobile browsers simply don't support frames. Besides, given the lack of pointing devices on most mobile devices, frames would result in a extremely poor user experience.

How?

Remove frames, and use a CSS-based layout instead.

Where?

Triggered by the resource under test:

Line 370

… <iframe allowtransparency="true" frameborder="0" scrolling="no" src="http://www.facebook.com/plugins/likebox.php?href=http%3A%2F%2Fwww.facebook.com%2Fmestradoecm&width=185&colorscheme=light&show_faces=true&border_color=%F2F2F2&stream=false&header=false&height=258" style="overflow: hidden; width: 300px; height: 270px;" />

More information

Severity: critical

Most mobile devices will not be able to render (part of) the page or will not be able to render the page in a reasonable time frame. Critical failures should be addressed first!

Category: Stay away from known hazards

Thoughtful design can help reduce usability problems due to small screens and keyboards, and other features of mobile devices.

Best practice:

Do not use frames.

De acordo com a recomendação da W3C a utilização do elemento iframe que corresponde à caixa de fãs do *Facebook* é considerada uma falha crítica no desenvolvimento do *website*, a prática recomendada no relatório será a de remover o elemento iframe.

O relatório divide o output da sua avaliação nas seguintes partes: *critical, severe, médium, low*. A ferramenta notifica que o output *critical* deve ser solucionado em primeiro lugar, e só depois se deve abordar os restantes. Estes parâmetros são úteis para entendermos aquilo que são as práticas recomendadas no desenvolvimento web compatível com dispositivos móveis. A razão do fraco desempenho do *website* nesta ferramenta de avaliação deve-se sobretudo ao facto de esta não considerar apenas *smartphones*, mas também telemóveis e PDAs, e o facto destes não apresentarem suporte ao HTML5.

Ainda a registar, o facto do m*edia querie* utilizado ser o *media screen* face ao típico *media handheld* recomendado pela W3C, prejudica os testes. Enquanto a última revisão dos media queries não for implementada não é possível sujeitar este tipo de *media screen* a esta plataforma de avaliação.

3.2 Universo Alvo Versus Universo Inquirido

A dificuldade de optar entre um universo alvo versus um universo inquirido para a realização de um questionário no âmbito da investigação realizada é minimizada pelos testes informatizados empregues. Segundo (Hill & Hill, 2005, p. 44), *"é melhor fazer uma boa investigação de âmbito limitado do que uma investigação fraca de grande escala e, normalmente, nem o tempo nem os recursos disponíveis que os alunos dispõem são adequados para fazer uma investigação de grande escala."* Observamos, à luz desta afirmação que optar por um universo inquirido faz sentido, ou seja, recorrer a uma amostragem dos utilizadores, com questões alternativas aos testes informatizados empregues reduzimos o tempo e evitamos a redundância na nossa investigação, em particular no input de dados e navegação do *website*.

3.3 Objetivos da investigação proposta

É o nosso principal objetivo com no âmbito da nossa investigação responder às seguintes questões:

- Quais as características do *design* dinâmico?
- Como implementar o *design* dinâmico num sistema de gestão de conteúdos?
- A introdução do *design* dinâmico em sistemas de gestão de conteúdos é satisfatória ao nível da experiencia do utilizador?

3.4 Recolha de dados por questionário

Através dos métodos descritos anteriormente, em termos de amostragem por universo inquirido, elaborámos o nosso questionário de acordo com o sugerido por Hill & Hill (2005, p. 163) "*é necessário estabelecer um compromisso entre a clareza do layout e o tamanho do questionário*". De facto, e em concordância com o autor, a importância do *layout*, não pode, nem deve ser subestimada, em função do seu objetivo primário que é a recolha de dados, a sua aparência estética não deve distrair, para que a cooperação entre o inquirido e o investigador não seja prejudicada.

3.5 Análise dos dados

O questionário teve como inquiridos o conjunto de docentes e discentes da área de tecnologias educativas da Escola Superior de Educação de Santarém, foi-lhes disponibilizada para consulta o *website* de MECM, desenvolvido no âmbito deste projeto. Justificamos a seleção da amostragem do universo inquirido, ou seja, da comunidade académica da Instituição, com o facto de que estes serão os utilizadores-alvo do *website* na sua fase de implementação.

O questionário foi enviado por *email* aos inquiridos com clara identificação e descrição do seu objetivo, a sua recolha foi realizada através do recurso: Survey Monkey, disponível em http://pt.surveymonkey.com.

A estrutura do questionário apresenta 10 questões dicotómicas relacionadas com a experiência de utilizador do *website* MECM. A 9˚ questão apresenta a possibilidade de resposta aberta dependendo da sua escolha.

3.5.1 Questões e respostas obtidas

1. Os URLs do *website* contêm as palavras-chave correspondentes ao seu conteúdo?

Respondidas: 7

Ignoradas: 0

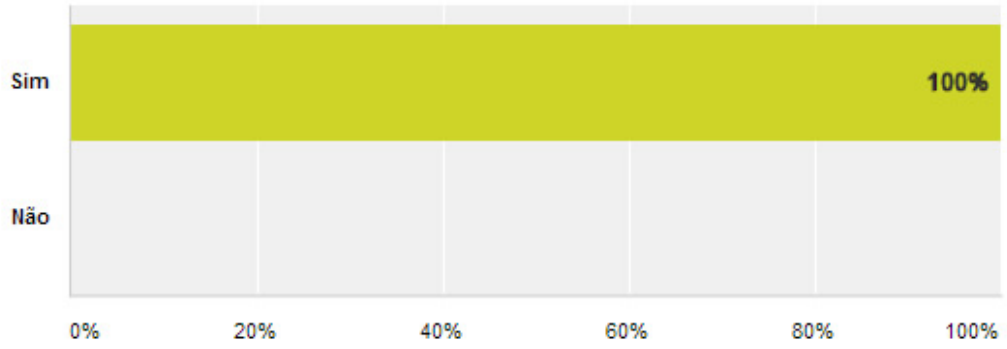


Gráfico 1 - Questão 1

Opções de resposta	Respostas
Sim	**100%** 7
Não	**0%** 0
Total	7

Esta questão tem como objetivo determinar se os termos empregues na construção dos URLs se encontram em concordância com os conteúdos apresentados. A totalidade dos inquiridos respondeu afirmativamente.

2. Acha adequado o tempo de carregamento do *website* de MECM no browser?

Respondidas: 7

Ignoradas:

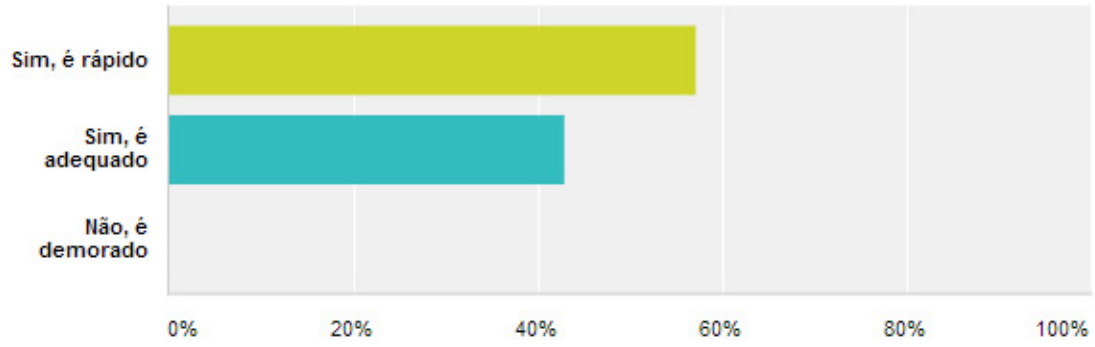


Gráfico 2 - Questão 2

Opções de resposta	Respostas
Sim, é rápido	**57,14%** 4
Sim, é adequado	**42,86%** 3
Não, é demorado	**0%** 0
Total	7

Relativamente ao tempo de carregamento do *website* no browser, 4 dos indivíduos respondem que é rápido e 3 que o tempo é adequado. Assim convém perceber as diferenças e condições no acesso ao *website* por parte dos inquiridos.

3. Considera a estrutura de navegação ao nível do menu principal adequada e percetível?

Respondidas: 7

Ignoradas: 0

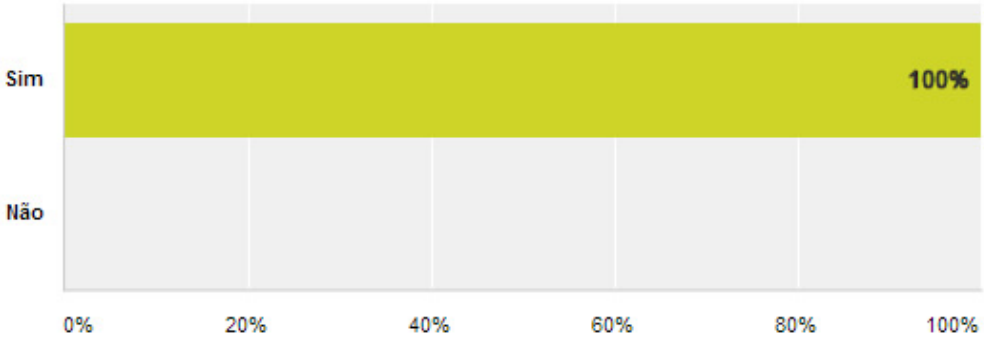


Gráfico 3 - Questão 3

Opções de resposta	Respostas
Sim	**100%** 7
Não	**0%** 0
Total	7

Esta questão pretende avaliar a perceção da estrutura de navegação do ponto de vista do utilizador. A totalidade dos inquiridos respondeu de forma afirmativa. Este feedback vai ao encontro dos resultados obtidos com o *Selenium* IDE ao nível da navegação.

4. Utiliza o Internet Explorer enquanto navegador web predefinido?

Respondidas: 7

Ignoradas: 0

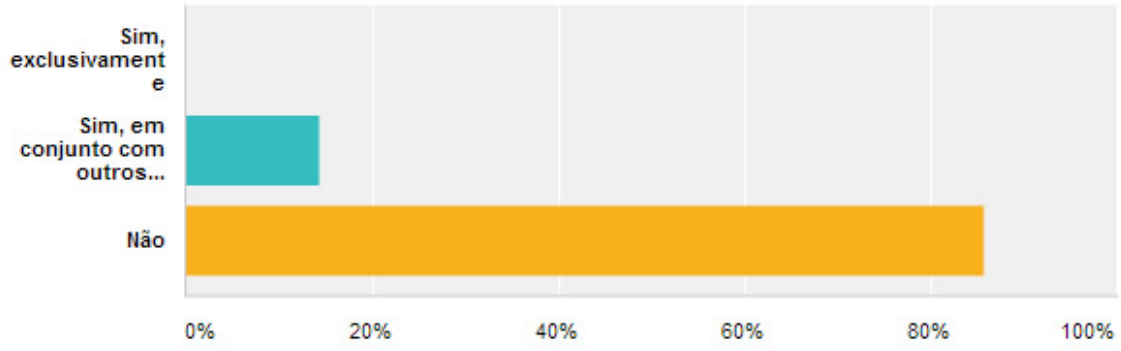


Gráfico 4 - Questão 4

Opções de resposta	Respostas
Sim, exclusivamente	**0%** 0
Sim, em conjunto com outros navegadores	**14,29%** 1
Não	**85,71%**

	6
Total	7

Esta questão pretende determinar a utilização do *browser* IE por parte dos utilizadores. De forma a compreender o resultado dos inquiridos na questão 2. Apenas 1 inquirido utiliza o Internet Explorer.

5. Se respondeu de forma afirmativa à questão anterior, qual é a versão do Internet Explorer que utiliza?

Respondidas: 1

Ignoradas: 6

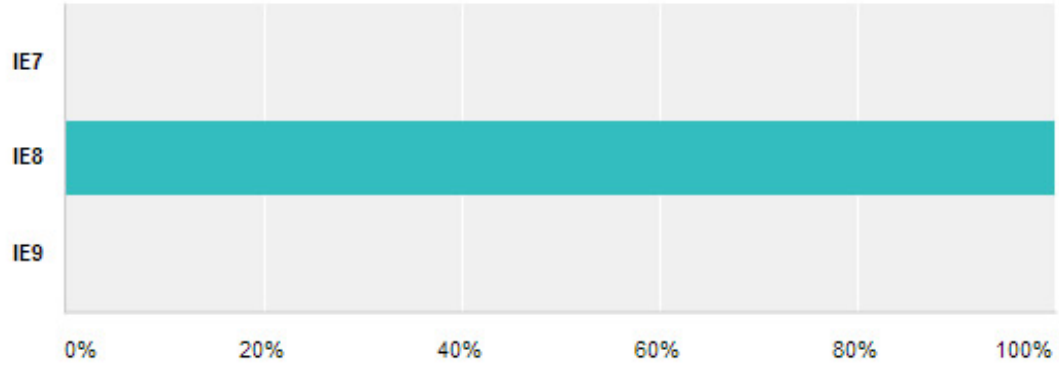


Gráfico 5 - Questão 5

Opções de resposta	Respostas
IE7	**0%** 0
IE8	**100%** 1
IE9	**0%** 0
Total	1

Esta resposta encontra-se condicionada à questão anterior, em que apenas um inquirido respondeu de forma afirmativa, logo apenas o mesmo inquirido poderia responder a esta questão. O objetivo desta questão é o de determinar o ambiente que determina a resposta à 2ª questão em função da *framework* escolhida para o desenvolvimento do projeto e da sua compatibilidade com as diferentes versões do IE.

6. Considera as diferenças do tamanho de letra utilizado no *website* proporcionais ao conteúdo, por exemplo, um tamanho menor para o corpo do texto, e um tamanho maior para os títulos das páginas?

Respondidas: 7

Ignoradas: 0

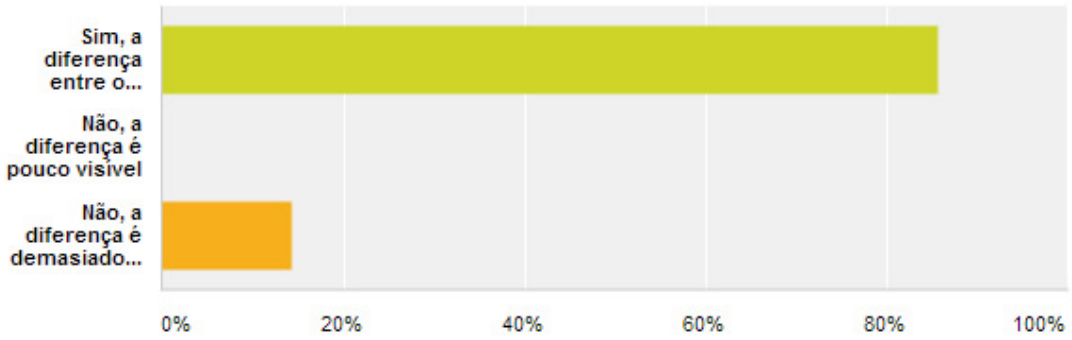


Gráfico 6 - Questão 6

Opções de resposta	Respostas
Sim, a diferença entre o tamanho das letras dos elementos de texto do *website* é adequada	**85,71%** 6
Não, a diferença é pouco visível	**0%** 0
Não, a diferença é demasiado acentuada	**14,29%** 1
Total	7

Esta questão refere-se à proporção entre o tamanho da letra do corpo do texto em relação ao tamanho da letra utilizado nos títulos. 6 dos inquiridos respondem de forma afirmativa, que consideram adequado a proporção do tamanho da letra, enquanto 1 inquirido responde que a diferença é demasiado acentuada. No global, a proporção do tamanho de letra é aceite pelos inquiridos.

7. O *design* dinâmico implementado neste *website* têm como principal objetivo acolher diferentes dispositivos, em concreto *smartphones* e *tablets-pc*. Acede habitualmente à Internet a partir de um *smartphone* ou *tablet-pc*?

Respondidas: 7

Ignoradas: 0

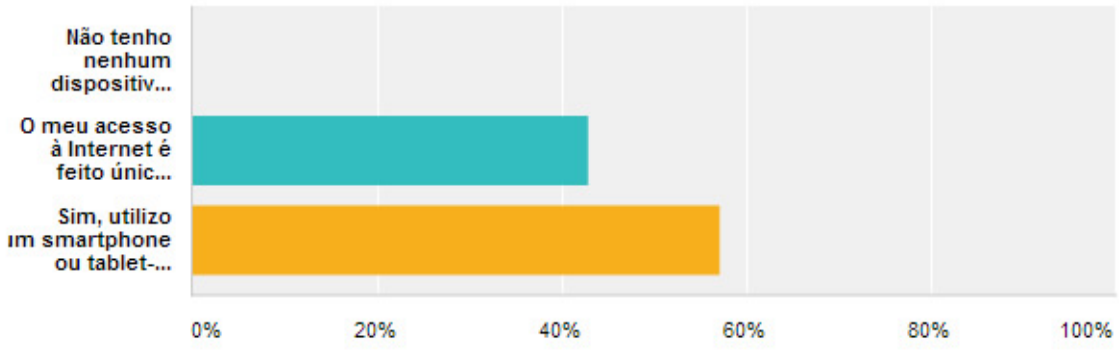


Gráfico 7 - Questão 7

Opções de resposta	Respostas
Não tenho nenhum dispositivo móvel	**0%** 0
O meu acesso à Internet é feito única e exclusivamente através de um *desktop*	**42,86%** 3
Sim, utilizo um *smartphone* ou *tablet-pc* para aceder à Internet com regularidade	**57,14%** 4
Total	7

Ao analisar o gráfico verificamos que a adoção de dispositivos móveis representa 57,14% , sensivelmente mais do que metade dos inquiridos utiliza um *smartphone* ou *tablet-pc* para aceder à Internet com regularidade.

8. No geral, em termos de experiência do utilizador, qual é o seu grau de satisfação com a utilização do *website* de MECM?

Respondidas: 7

Ignoradas: 0

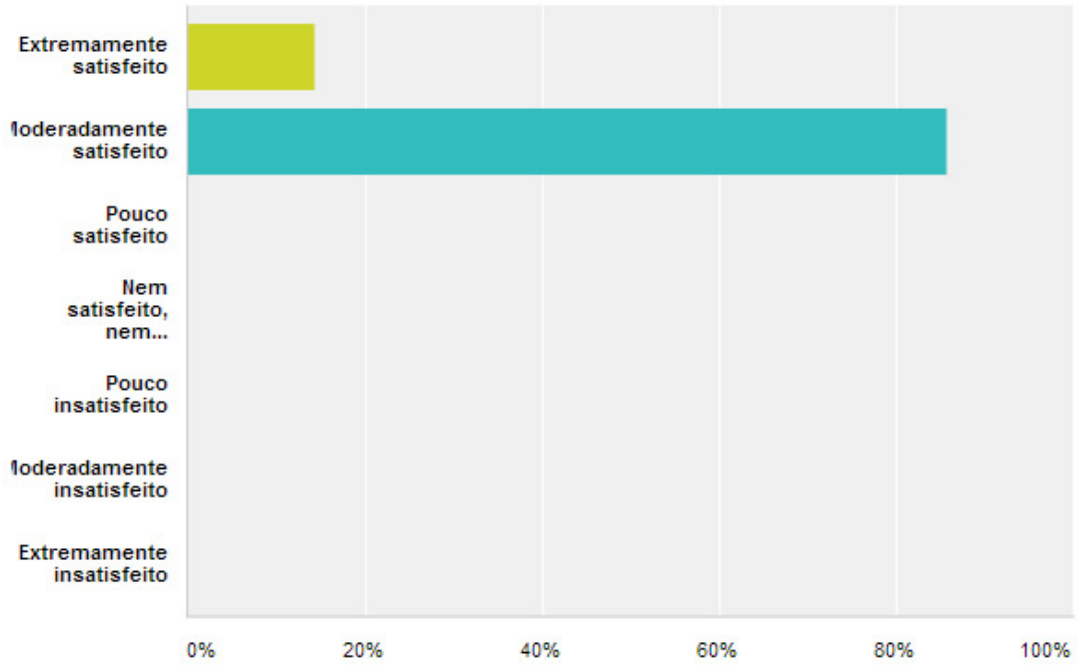


Gráfico 8 - Questão 8

Opções de resposta	Respostas
Extremamente satisfeito	**14,29%** 1
Moderadamente satisfeito	**85,71%** 6
Pouco satisfeito	**0%** 0
Nem satisfeito, nem insatisfeito	**0%** 0
Pouco insatisfeito	**0%** 0
Moderadamente insatisfeito	**0%** 0
Extremamente insatisfeito	**0%** 0
Total	7

A maioria dos inquiridos, 6, apresentam-se extremamente satisfeitos com a utilização do *website*, um inquirido apresenta-se moderadamente satisfeito.

9. Experimenta alguma dificuldade na consulta dos conteúdos do *website* de MECM a partir de um *smartphone* ou *tablet-pc*?

Respondidas: 7

Ignoradas: 0

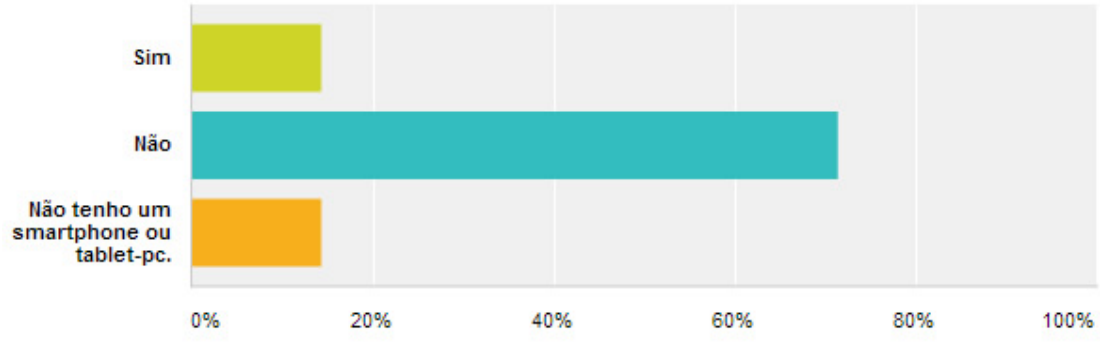


Gráfico 9 - Questão 9

Opções de resposta	Respostas
Sim	**14,29%** 1
Não	**71,43%** 5
Não tenho um *smartphone* ou *tablet-pc*.	**14,29%** 1
Total	7
Em caso afirmativo especifique: (1)	

(1)Existe alguma dificuldade se operar o smartphone na posição vertical (portrait), se visualizar na horizontal (landscape) tal já não sucede, contudo tenho de recarregar a página quando coloco na horizontal pois os conteúdos "perdidos" na posição vertical não surgem automaticamente.

Quanto à dificuldade na consulta dos conteúdos do website de MECM, um dos inquiridos responde que não tem nenhum *smartphone* ou *tablet-pc*, 5 não tiveram dificuldade, e 1 experienciou dificuldades e justificou a sua escolha. Significa que um dos inquiridos tem um *smartphone* com resolução de ecrã inferior a 320 pixéis, a solução passa por conceber um @media *screen* para medidas inferiores a 320 pixéis.

10. Considera que a implementação do *design* dinâmico é uma necessidade no desenvolvimento *Web*?

Respondidas: 7

Ignoradas: 0

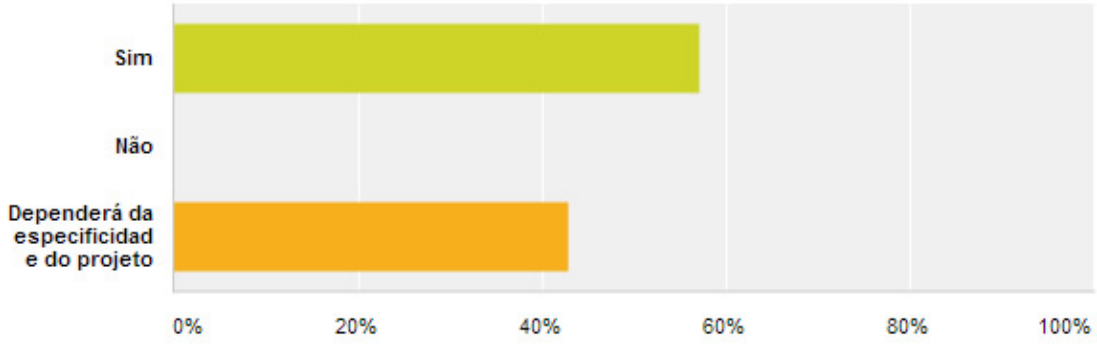


Gráfico 10 - Questão 10

Opções de resposta	Respostas
Sim	**57,14%** 4
Não	**0%** 0
Dependerá da especificidade do projeto	**42,86%** 3
Total	7

Relativamente à necessidade de implementação do *design* dinâmico no desenvolvimento *Web*, 4 dos inquiridos respondem que sim e 3 que dependerá da especificidade do projeto.

Conclusão

Este projeto foi desenvolvido em grande parte com a bibliografia apresentada como base, em conjunto com a documentação oficial do *Wordpress* e com a aplicação em simultâneo das recomendações oficiais da W3C. Numa breve observação, estas recomendações foram implementadas de acordo com as necessidades do projeto em particular, contudo num determinado número de situações essa implementação iria numa direção oposta aquela que pretendemos, em concreto, uma experiencia de navegação adequada a dispositivos móveis. Na página de publicações em I&D, tradicionalmente esse tipo de conteúdos extensos seria apresentado em índice com as devidas ligações, considerando a área de ecrã de um *smarphone*, a situação iria resultar numa navegação não linear. Decidimos utilizar elementos *CSS* e funções em *Javascript* para ocultar e expandir as publicações ordenadas por data, facilitando a usabilidade deste conteúdo específico para os dispositivos móveis. A consequência desta abordagem resulta num erro de validação do código HTML5, pelos instrumentos de avaliação da W3C, este sugere a implementação do índice, contudo esta alternativa afeta a usabilidade do *website* em dispositivos móveis com ecrãs de menor dimensão, o uso desta exceção foi necessária, ainda que não recomendado pela W3C – consultar testes de validação e acessibilidade em anexo.

O principal obstáculo à abordagem RWD proposta encontra-se no acesso 3G, ao pretendermos um único documento responsável pelos *media queries* ou pela apresentação do mesmo *layout* numa multiplicidade de dispositivos alvo, temos de considerar que a totalidade da informação e conteúdos no website é carregada no dispositivo-alvo independente das suas capacidades de processamento e qualidade de acesso à Internet. De momento a tecnologia disponível ao consumidor-utilizador no dispositivo-alvo limita os *media queries* ao rácio e orientação de ecrã. Apesar da introdução de novos *media queries* se encontrar prevista para breve pela, levará ainda algum tempo para a implementação desses elementos. Em primeiro lugar os motores de rendering de interface dos navegadores atuais, o *Trident* do *Internet Explorer*, o *Gecko* do *Mozilla Firefox*, o *Webkit* que é comum ao *Safari* e *Chrome* não preveem esses elementos, e claro está, há que considerar as diversas adaptações destes navegadores nos diversos sistemas operativos e dispositivos móveis. Mas acima de tudo temos de considerar a introdução destes elementos pelas entidades e indivíduos responsáveis pela produção programática do código e do *design*. A introdução de *media queries* específicos para largura de banda no acesso à Internet (Coyer, Cannon, & Stewart, 2012, p. 10) seria uma solução directa à problemática do acesso 3G em dispositivos móveis a websites produzidos com técnicas de RWD.

E porque não introduzir *media queries* para interpretar as especificações do CPU e RAM, este elemento combinado com a proposta de *Coyier*, permitiria filtrar o conteúdo de um *website*, em concreto conteúdos do tipo imagens e vídeos ao disponibilizar no dispositivo alvo uma mensagem de texto alternativa através do elemento HTML de acessibilidade <alt> que não é possível apresentar o conteúdo devido à limitação no acesso à Internet via 3G, e da capacidade de processamento do dispositivo-alvo. Uma afirmação comum aos autores presentes na bibliografia do presente relatório é a de desconhecerem as condições em que os utilizadores acedem aos conteúdos, da qualidade da largura de banda,

móvel, cabo, ADSL ou fibra ótica, e das características específicas do dispositivo-alvo. É nossa firme posição com base na investigação registada neste relatório e na implementação do projeto da fase de desenvolvimento para a fase de produção que a evolução do *design* dinâmico na Web passa pela introdução de novos *media queries*.

Observemos por outro lado a evolução do sistema de gestão de conteúdos, ou melhor o seu ciclo de vida enquanto *software*, e neste caso não nos vamos limitar com o seu âmbito, ou tipologia, se gestão documental, ensino a distância, comércio eletrónico, blog ou qualquer outro género de publicação de conteúdos. Em cada nova versão, há novas funcionalidades e formas de expandir as funcionalidades existentes com APIs. Da mesma forma que o *WordPress*, *Drupal* e *Moodle* apresentam bibliotecas de *javascript* integradas para o interface, como jQuery e YUI (Yahoo User Interface), podemos assumir a viabilidade de introdução de bibliotecas CSS, na forma de *framework* ou *preprocessor* para o desenvolvimento de APIs com funções nativas de *media queries*. O que levanta a questão, deve o desenvolvimento de *design* dinâmico em CMS ser uma funcionalidade integrada, nativa desse tipo de sistemas? Essa hipótese é na nossa opinião, formada através desta investigação uma possibilidade a desenvolver.

O elevado custo no desenvolvimento e manutenção de aplicações nativas em sistemas operativos móveis, como o iOS, *Android*, *Windows Mobile*, *Blackberry* entre outros faz do desenvolvimento e implementação do RWD em aplicações *web* uma alternativa barata mas eficaz na distribuição de conteúdos. Em que a aplicação vai além de *websites* estáticos e até mesmo sistemas de gestão de conteúdos, a sua recente integração no mercado dos *ebooks* é a confirmação desta tecnologia, quer seja no âmbito educacional ou em contexto económico. Concluímos também que a tecnologia RWD é um importante contributo na reutilização do código de *scripting* utilizado na web - HTML, CSS e Javascript e na conversão deste em aplicações nativas, com recursos a *frameworks.* Como o *PhoneGap*, que converte o código tipicamente utilizado na produção de um *website* ou aplicação *web* em aplicações para Android e iOS, com os devidos ajustes através das APIs fornecidas. Recentemente com o aparecimento do Firefox OS e Ubuntu Phone a introdução do código *web* para a produção de aplicações surge integrada no núcleo do SDK disponibilizado para programadores. Esta diminuição nos custos de produção de aplicações para plataformas móveis é uma consequência direta da introdução da tecnologia RWD , resta determinar o peso da sua influência na economia digital.

Em análise dos dados bibliográficos e da relação espaço temporal dos mesmos podemos também concluir que a discussão da tecnologia RWD por parte da comunidade científica constituída por programadores, *Web designers*, investigadores, instituições e empresas parceiras da W3C foi transversal às necessidades do mercado das operadoras móveis e fabricantes de dispositivos em tempo útil. Temos uma forma de *web design* que atende às necessidades da economia digital. Mais interessante ainda é a forma como esta tecnologia é implementada em diferentes regiões do globo com elevado nível de aceitação, é difícil abster-nos da representação e relação das camadas geográficas da *World Wide Web* (Castells, 2004). Segundo o autor, o trajeto da sociedade e sustentabilidade da economia será realizado através do desenvolvimento científico e tecnológico, em ambientes digitais colaborativos na *Internet*. A adoção massiva do *design* dinâmico no desenvolvimento *Web* facilita esses ambientes colaborativos em dispositivos ou equipamentos móveis como o *smartphone*.

Esperamos com este trabalho levar o leitor a perceber a necessidade de implementação do *design* de interfaces dinâmicos em sistemas de gestão de conteúdos, é a nossa opinião com base na investigação deste documento que esta é uma etapa fundamental no ciclo de vida deste tipo de software, mas acima de tudo na forma como interagimos com os conteúdos através dele, sobretudo na área da experiência e design do utilizador. Assim sendo deixamos esta frase enquanto fundamento e elemento percursor do tema do presente relatório: "*El mundo se hace móvil y las nuevas generaciones, interactivas, y desde las instituciones educativas debemos hacer un esfuerzo por integrar y normalizar y no por excluir estas potenciales herramientas educativa"* (Brazuelo Grund & Cacheiro González, 2010, p. 11).

Bibliografia

Berjon, R. -W., Leithead, T., Navara, Microsoft, E. , O'Connor, E. -A., Pfeiffer, S., et al. (7 de Dezembro de 2012). *HTML5 Definition Complete, W3C Moves to Interoperability Testing and Performance*. Obtido em 3 de Janeiro de 2013, de W3C For Immediate Release: http://www.w3.org/2012/12/html5-cr

Betts, Ryan - Adobe Systems; Lillesveen, Rune; Stenhaug, Øyvind - Opera Software; Rivoal, Florian. (1 de março de 2013). *CSS Device Adaptation.* Obtido em 4 de Maio de 2013, de W3C Editor's Draft : http://dev.w3.org/csswg/css-device-adapt/

Brazuelo Grund, F., & Cacheiro González, M. L. (2010). Diseño de páginas web educativas para teléfonos móviles. *Edutec: Revista electrónica de tecnología educativa. Issue 32. Fundación Dialnet.*

Castells, M. (2004). *A Galáxia Internet: Reflexões sobre Internet, Negócios e Sociedade.* Lisboa: Fundação Calouste Gulbenkian.

Christian, P., & Beale, R. (2008). *Affect and Emotion in Human-Computer Interaction: From Theory to Applications.* Springer Science & Business Media B.V.

Codex. (s.d.). *Codex*. Obtido de WordPress: http://codex.wordpress.org/

Connors, Adam; Google. (14 de Dezembro de 2010). *Mobile Web Application Best Practices.* Obtido em 4 de Novembro de 2012, de W3C Recommendation: http://www.w3.org/TR/mwabp/#bp-viewport

Coyer, C., Cannon, S., & Stewart, I. (2012). *Wordpress meet Responsive Design.* Code Poet.

Göktürk, M. (2008). *Fitts's Law*. Obtido em 12 de 11 de 2012, de Interaction Design: http://www.interaction-design.org/encyclopedia/fitts_law.html

Grudin, J. (2011). Human-Computer Interaction. *Annual Review Of Information Science And Technology - Vol.45*, pp. 369-430.

Hill, M., & Hill, A. (2005). *Investigação Por Questionário.* Lisboa: Edições Sílabo.

Jones, K., & Farrington, P.-A. (2011). *Using Wordpress As a Library Content Management System.* American Library Association.

Keith, J. (2005). *DOM Scripting Web Design with JavaScript and the Document Object Model.* Springer Science & Business Media B.V.

Lie, Håkon; Rivoal, Florian; Van Kesteren, Anne - Opera Software; Çelik, Tantek - Stanford; Glazman, Daniel - Desruptive Innovations. (19 de Junho de 2012). *Media Queries.* Obtido em 17 de Dezembro de 2012, de W3C Recommendation: http://www.w3.org/TR/css3-mediaqueries/

Maragos, P., Potamianos, A., & Gros, P. (2008). Multimedia Systems and Applications Volume 33. *Multimodal Processing and Interaction: Audio, Video, Text*, pp. 1-39.

Marcotte, E. (25 de Maio de 2010). *Responsive Web Design*. Obtido em 8 de Outubro de 2012, de A List Apart: http://alistapart.com/article/responsive-web-design

Rabin, J. -D., & McCathieNevile, C. -O. (29 de Julho de 2008). *Mobile Web Best Practices 1.0.* Obtido em 19 de Dezembro de 2013, de W3C Recommendation: http://www.w3.org/TR/mobile-bp/#OneWeb

Schmidt, A., & Etches, J. A. (2011). *Ten ways WordPress can improve website UX.* Cengage Learning, Inc.

Spinellis, D. (Maio de 2012). Git. *IEEE Software Vol. 29 Issue 3. EBSCO.*, pp. 100-101.

Spitzer, S. (Junho de 2012). Re-engineering a web portal for mobile access. *Computers in Libraries Vol.32* , p. 10.

Stern, H., Damstra, D., & Williams, B. (2010). *Professional WordPress.* Wiley Publishing, Inc.

Vance, A. (2 de Novembro de 2009). *Drupal Moves Into the White House*. Obtido em 20 de Novembro de 2012, de The New York Times: http://bits.blogs.nytimes.com/2009/10/30/drupal-moves-into-the-white-house/

Anexos

Configuração de permissões e acesso a ficheiros e pastas no *Wordpress*.

Numero	Representação de permissões	Parâmetros
0	Sem permissão	---
1	Executar	--x
2	Escrever	-w-
3	Executar e escrever: 1 (executar) + 2 (escrever) = 3	-wx
4	Ler	r--
5	Ler e executar: 4 (ler) + 1 (executar) = 5	r-x
6	Ler e escrever: 4 (ler) + 2 (escrever) = 6	rw-
7	Todas as permissões: 4 (ler) + 2 (escrever) + 1 (executar) = 7	rwx

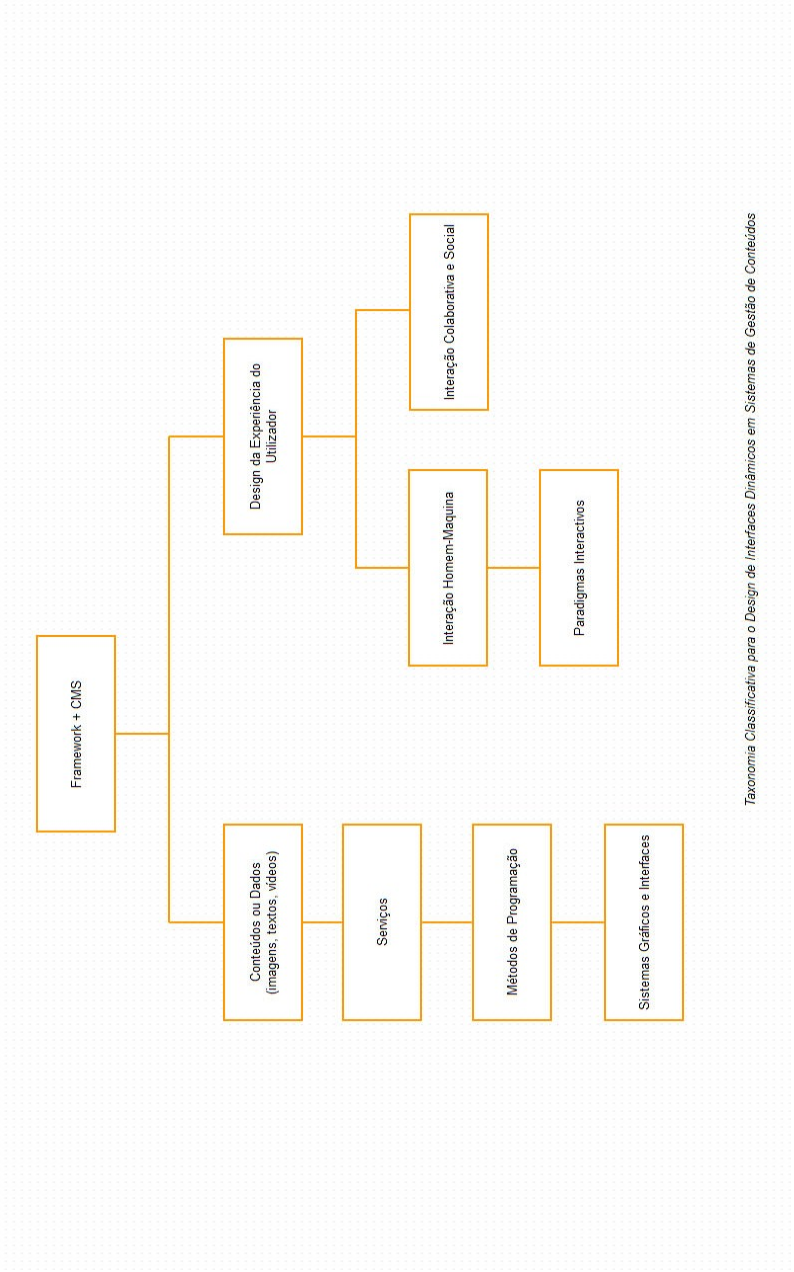
Tabela 1 - Tabela representativa de valores de permissões para acesso a ficheiros e pastas.

Segundo a tabela 1, e de acordo com as recomendações oficias do códex do *Wordpress* estas são as configurações apropriadas a atribuir aos ficheiros:

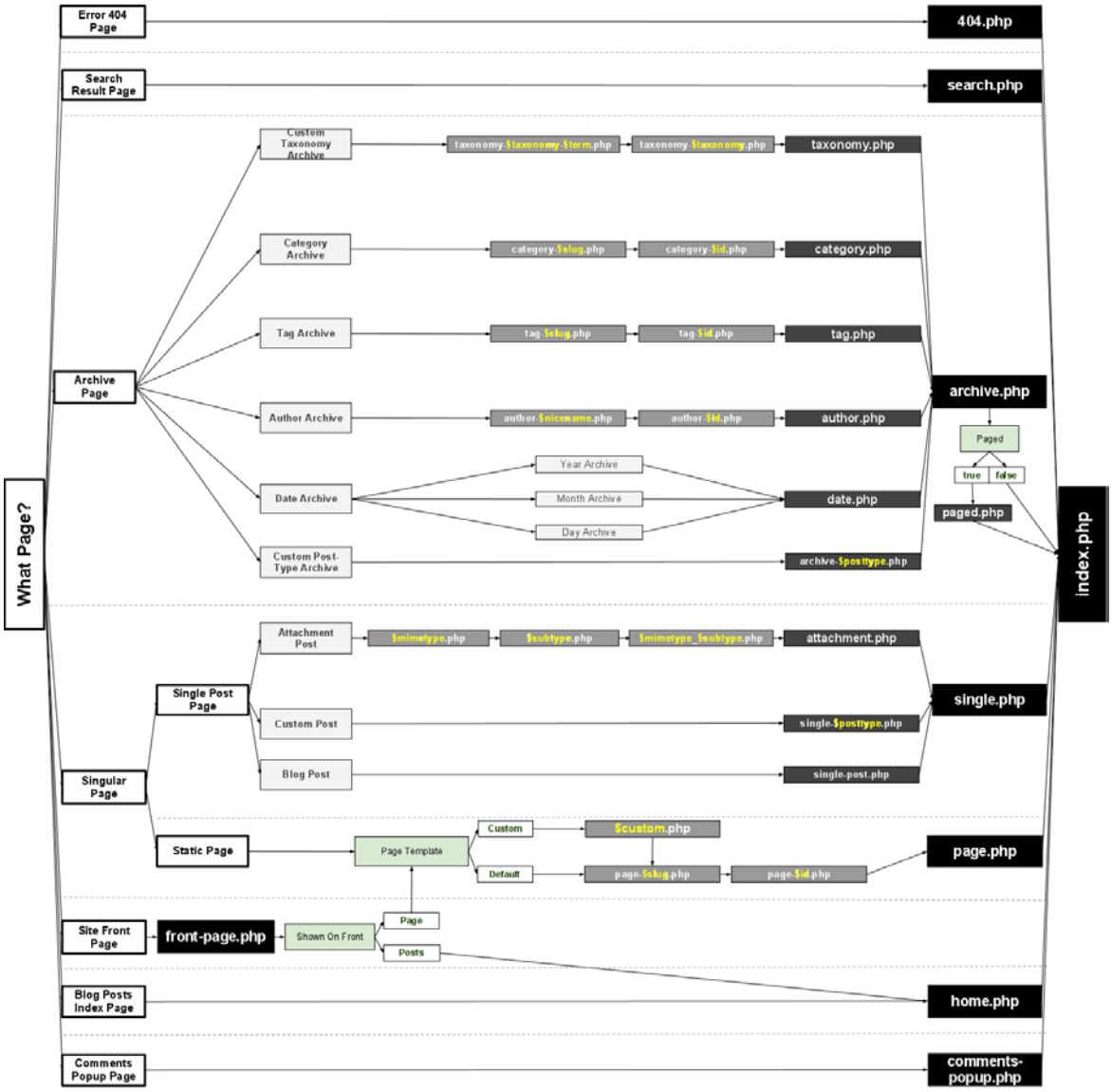
- **644**: Podem ser lidos por todos e executados apenas pelo proprietário do arquivo.
- **755**: Ler, escrever e permissão de execução para o proprietário. Todos os outros podem ler e executar o arquivo, mas não estão autorizados a modificar ou apagar.

Em caso algum se deve atribuir o valor 777 aos ficheiros ou pastas, esse valor traduz-se em permissões absolutas para ler, escrever e executar, ou seja todos os utilizadores teriam acesso sem restrições aos ficheiros de sistema do CMS, o que representa um grave compromisso de segurança.

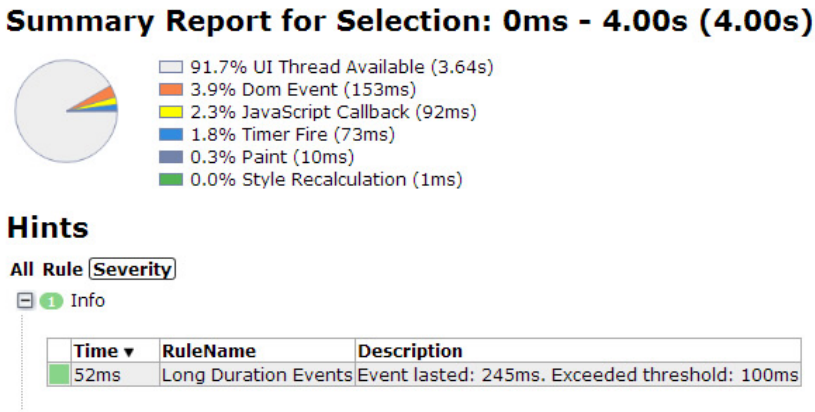
Taxonomia Classificativa para o Design de Interfaces Dinâmicos em Sistemas de Gestão de Conteúdos



Vista comum de ficheiros de *template* interpretados pelo *Wordpress*



Performance do *Website*



Acesso à Internet em banda larga móvel

5.2.5 Penetração do Serviço Acesso à Internet em banda larga móvel / *Internet Mobile Broadband Access Market Penetration*

	2005	2006	2007	2008	2009	2010	2011
Percentagem de clientes de Banda Larga (móvel) activos por 100 habitantes / Active Mobile Broadband customers per 100 inhabitants			6.2	10.9	20.4	24.1	27.5

Fonte/Source: ICP-ANACOM

Unidade/Unit: Nº clientes/100 habitantes
Number of customers per 100 inhabitants

Notas/Notes:

Trata-se dos clientes dos operadores móveis que podem aceder à Internet em banda larga móvel, e que estabeleceram pelo menos uma sessão IP para acesso à Internet em banda larga, no período de reporte, ie registaram tráfego no último mês do trimestre. Corresponde ao indicador 2.5.1.1 do Questionário trimestral dos serviços móveis. Consultar a definição deste indicador no sítio da ANACOM, no endereço http://www.anacom.pt/render.jsp?contentId=963861 (Página Inicial > Estatísticas > Operadores / prestadores - informação periódica a remeter à ANACOM > Questionários trimestrais por serviço > Serviços Móveis > Serviços Móveis - Deliberação de 17.06.2010 (para envio a partir do 2.º trimestre 2010 e até 30 Julho 2010)).

Esta informação foi fornecida pelos prestadores do serviço e poderá ser objecto de correcções no futuro/*These figures were based on data supplied by service providers and may be corrected.*

Questionário

Questionário para avaliação do *design* dinâmico e experiência de utilizador do *website* do Mestrado de Educação e Comunicação Multimédia, disponível para consulta em http://mecm.eraizes.com.

O questionário é composto por 10 questões. Cada questão apresenta várias opções de resposta, apenas pode escolher uma resposta por questão. A consulta do *website* em http://mecm.eraizes.com é obrigatória para o preenchimento do questionário.

É o objetivo deste questionário recolher as impressões gerais acerca das tecnologias nele empregues por parte de uma amostragem de utilizadores.

Este questionário foi elaborado no âmbito da unidade curricular de Seminário de 2° ano, 2° semestre do Mestrado de Educação e comunicação Multimédia da Escola Superior de Educação de Santarém.

Tempo previsto para a duração do questionário: 5 minutos.

1. Os URLs do *website* contêm as palavras-chave correspondentes ao seu conteúdo?

- Sim
- Não

2. Acha adequado o tempo de carregamento do website de MECM no browser?

- Sim, é rápido
- Sim, é adequado
- Não, é demorado

3. Considera a estrutura de navegação ao nível do menu principal adequada e perceptível?

- Sim
- Não

4. Utiliza o Internet Explorer enquanto navegador web predefinido?

- Sim, exclusivamente
- Sim, em conjunto com outros navegadores
- Não

5. Se respondeu de forma afirmativa à questão anterior, qual é a versão do Internet Explorer que utiliza?

- IE7
- IE8

- IE9

6. Considera as diferenças do tamanho de letra utilizado no site proporcionais ao conteúdo, por exemplo, uma tamanho menor para o corpo do texto, e um tamanho maior para os títulos das paginas ?

- Sim, a diferença entre o tamanho das letras dos elementos de texto do site é adequada
- Não, a diferença é pouco visível
- Não, a diferença é demasiado acentuada

7. O design dinâmico implementado neste website têm como principal objetivo acolher diferentes dispositivos, em concreto smartphones e tablets-pc. Acede habitualmente à Internet a partir de um smartphone ou tablet-pc?

- Não tenho nenhum dispositivo móvel
- O meu acesso à Internet é feito única e exclusivamente através de um desktop
- Sim, utilizo um smartphone ou tablet-pc para aceder à Internet com regularidade

8. No geral, em termos de experiência do utilizador, qual é o seu grau de satisfação com a utilização do site de MECM?

- Extremamente satisfeito
- Moderadamente satisfeito
- Pouco satisfeito
- Nem satisfeito, nem insatisfeito
- Pouco insatisfeito
- Moderadamente insatisfeito
- Extremamente insatisfeito

9. Experimenta alguma dificuldade na consulta dos conteúdos do website de MECM a partir de um smartphone ou tablet-pc?

- Sim
- Não
- Não tenho um smartphone ou tablet-pc.

Em caso afirmativo especifique:

10. Considera que a implementação do design dinâmico é uma necessidade no desenvolvimento web?

- Sim
- Não
- Dependerá da especificidade do projeto

Visualização do *website* de MECM em diferentes dispositivos