Resumos LTW

MIEIC

11 de fevereiro de 2021

Conteúdo

1 Introdução **2**

2 HTML **2**

3 CSS **3**

4 PHP **4**

5 JavaScript **6**

6 Web Security **8**
- 6.1 Path Traversal Attack . . . 8
- 6.2 SQL Injection . . . 8
- 6.3 Account Lockout . . . 8
- 6.4 Cross-Site Scripting (XSS) . . . 9
- 6.5 Cross-Site Request Forgery (CSRF) . . . 9
- 6.6 Man in the Middle Attack . . . 9
- 6.7 Credential Storage . . . 9
- 6.8 Session Fixation . . . 9
- 6.9 Session Hijacking . . . 9
- 6.10 Denial of Service . . . 9

7 Regular Expressions **9**

8 HTTP **12**

9 XML **13**

10 XPath **14**

1 Introdução

Este documento contém resumos dos conteúdos lecionados em LTW no ano letivo 2019/2020. Este documento contém apenas resumos dos conteúdos abordados na unidade curricular e não substitui o estudo mais aprofundado destes conteúdos.

2 HTML

Alguns elementos HTML podem ter filhos. Estes elementos têm tag de abertura e fecho. Elementos que não permitem ter filhos só têm tag e abertura.

Um documento HTML tem uma tag html como raíz e as secções head (que tem que conter a tag title) e body. A tag html deve conter o atributo lang.

Listas:

- Ordenadas (ol).
- Não ordenadas (ul).
- Descritivas (dl): definem-se termos (dt) e as suas descrições (dd).

Tabelas (table):

- Pode ter um título/legenda (caption).
- Organizada através de linhas (tr) que contêm células de dados (td).
- Algumas células podem ser cabeçalhos (headers): th em vez de td.
- colspan: célula ocupa número de colunas especificado.
- rowspan: célula ocupa número de linhas especificado.
- Secções: thead, tfoot, tbody: permite especificar header, footer e body de uma tabela.
- Grupos de colunas: para não ser repetida informação nas colunas, definem-se grupos de colunas com as tags colgroup e col.

Formulários (form):

- Dois atributos importantes
  - action: página para onde serão enviados os resultados.
  - method: get ou post.
- Controlos:

  - input: campos editáveis.
  - textarea: campo de texto editável.
  - select: dropdown list.
  - button: botão genérico.
- Select (dropdown list):
  - Cada opção é representada com a tag option.
  - Opções podem ser agrupadas com optgroup (atributo label indica o nome do grupo).
- Label: permite associação entre uma label e o respetivo input. Clicando na label o input é ativado.
- Field Set (fieldset): permite agrupar inputs. Tag legend contém o título do grupo.

Um documento HTML não pode ter dois elementos com o mesmo id.

Elementos semânticos (alternativa a div): header, nav, aside, section, article, footer.

3 CSS

Seletores: permitem selecionar elementos HTML a serem alterados.

Propriedades: aspetos a alterar nos elementos selecionados.

<link rel="stylesheet" href="style.css">: ligar ficheiro HTML ao ficheiro *style.css*.

>: seleciona "filhos".

+: seleciona próximo "irmão".

~: seleciona próximos "irmãos".

Pseudo-classes permitem selecionar elementos com determinadas características.

Seletores de atributos: seleciona elementos baseando-se na existência de atributos e valores (exemplo: form[method=get].

Tamanhos relativos de fontes:

- **em**: quando usado com font-size, representa o tamanho da fonte do elemento-"pai". Para comprimentos, representa o tamanho da fonte do elemento atual.

Flexbox: alternativa ao box layout model. Elementos são flexíveis e adaptam-se ao tamanho atual do ecrã. É necessário alterar o atributo display para flex.

Grid: permite alinhar elementos em linhas e colunas.

Cálculo de especificidade:

- A especificidade de uma regra é definida como 4 valores (a, b, c, d).
- Cada valor é incrementado quando um certo tipo de seletor é utilizado:
  - d: Elemento, Pseudo-elemento (::).
  - c: Classe, Pseudo-classe (:), Atributo.
  - b: id.
  - a: Inline style.

CSS Vars:

- Entidades que podem ser reutilizadas no documento.
- Acedem-se com a função var().

4 PHP

Código PHP tem que ser delimitado através de "<?php"e "?>", ou "<?"e "?>", ou "<script language="php">"e "</script>".

Imprimir strings:

- <?php echo 'string'; ?>
- <?= 'string' ?>

Variáveis são representadas com $ seguido do nome da variável.

A atribuição a uma variável é feita por valor exceto se o sinal & for usado.

O valor null representa uma variável sem valor.

Uma variável é nula (null) se:

- lhe foi atribuída a constante null.
- ainda não lhe foi atribuído um valor.
- foi utilizada a função unset().

As funções die e exit param a execução de um script PHP. Podem receber um argumento (status).

Dois tipos de comparação:

- == e !=: comparam-se valores depois de haver cast para o mesmo tipo de variável.
- === e !==: comparam-se valores verdadeiros.

Strings podem ser definidas com aspas ou plicas. Strings com aspas expandem as variáveis que se encontram dentro.

Concatenação de strings é feita com o operador `.`

Arrays:

- Podem ser criados com a função array() ou [].
- São mapas organizados como coleção de pares key-value.
- Por defeito, as chaves são inteiros. Quando uma chave não é especificada, o PHP usa o incremento da maior chave inteira utilizada até ao momento.

Para utilizar uma variável global dentro de uma função é necessário declará-la como global.

Classes:

- Declaradas com class.
- Podem ter atributos/métodos privados, protegidos ou públicos.
- Para utilizar um atributo num méto método é utilizada a keyword $this.
- Podem ser declarados constructor (__contstruct()) e destrutor (__destruct()).
- Uma instância da classe é criada com a keyword new.
- Existe herança entre classes (extends) e é possível implementar interfaces (implements).

Para conectar a uma base de dados é utilizado um objeto PDO.

Para evitar ataques:

```
$stmt = $db->prepare(query);
$stmt->execute(array(args));
$stmt->fetch(); // Retira um resultado (fetchAll() retorna todos)
```

$_GET[] e $_POST[] permitem receber argumentos passados por estes métodos.

$_COOKIE permite aceder aos cookies enviados pelo browser.

Uma sessão é começada com a função session_start(). A variável $_SESSION guarda um array com as informações importantes da sessão. A função session_destroy() destrói todos os dados associados à sessão atual.

5 JavaScript

Para usar ECMAScript 5, o ficheiro de JavaScript deve começar com 'use strict'.

- Alterações:
  - Não existem variáveis globais não declaradas.
  - Não se declaram variáveis com var.
  - Alguns warnings passaram a erros.

Variáveis são declaradas com let. Os seus nomes podem ter apenas números, letras, $ e _ (e não podem começar com um número).

Constantes são declaradas com const.

Variáveis declaradas com var:

- Não têm block scope (só function scope).
- São processadas quando uma função começa.

Tipos de dados primitivos:

- Number (double).
- String (text).
- Boolean (true ou false).
- Null (apenas 1 valor: null).
- Undefined (ainda não foi atribuído um valor).

Strings podem ser definidas com plicas, aspas ou backticks (‘). Com a última, expressões dentro de ${...} são avaliadas e o resultado faz parte da string.

Operador + soma números ou concatena strings (se pelo menos um operando for uma string).

É possível usar String(), Boolean() e Number() para converter valores para os tipos pretendidos.

=== e !== comparam valores. == e != comparam tipos.

Funções são definidas com a keyword function. Tipos primitivos são passados por valor, mas tipos não primitivos são passados por referência. Funções com um return vazio ou sem return, retornam undefined.

Arrow functions são uma forma mais compacta de declarar funções.

Pode-se usar a keyword this dentro de um objeto para referir ao objeto atual.

Call e Apply são alternativas a chamadas a funções. Ambas recebem o contexto como primeiro argumento.

Objetos podem ser acedidos com [] como num array associativo.

for ... in permite executar código para cada propriedade de um objeto/array.

Cada função tem uma propriedade prototype interna que é inicializada como um objeto vazio. Quando o new é utilizado, é criado um novo objeto derivado do prototype. É possível alterar o prototype de uma função deretamente.

Quando um objeto é criado com new, uma propriedade __proto__ é inicializada com o prototype da função que o criou.

A keyword class é uma forma simplificada de criar classes protoype-based. Só podem ter métodos e getters/setters.

Arrays são objetos list-like. Só podem ser acedidos com []. Os seus índices começam em 0.

Podem ser lançadas exceções (ou qualquer outro objeto) com throw.

Exceções devem ser envolvidas em blocos try .. catch.

JavaScript pode ser incluído num ficheiro HTML com a tag script.

Scripts devem ter 1 dos seguintes métodos:

- async: é corrido assim que seja feito o seu download sem bloquear o browser.
- defer: é corrido apenas quando a página é carregada e por ordem.

document representa o documento atual. Contém métodos importantes para aceder a elementos. Um objeto Element representa um elemento HTML (id, innerHTML, outerHTML, style, getAttribute(), setAttribute(), remove()). A função createElement() de document cria um novo elemento, mas não é inserido no documento. O objeto Node representa um nó na árvore do documento (appendChild(), replaceChild(), removeChild(), insertBefore()).

XMLHttpRequest permite enviar pedidos HTTP facilmenteL

- open(method, url, async)
  - Method: get ou post.
  - URL: URL a procurar.
  - Async: se for falso, a execução para à espera de uma resposta.
- encodeForAjax(obj): cria um objeto a passar ao PHP.
- send(): envia os dados.
- JSON.parse(): permite receber a resposta do servidor.

6 Web Security

6.1 Path Traversal Attack

Utilização de .. e / para aceder a ficheiros e diretórios que não devem ser acedidos.

6.2 SQL Injection

Injeção e queries SQL a partir de dados de input.

6.3 Account Lockout

A aplicação contém bloqueio de contas que pode ser ativado facilmente (mecanismo usado contra ataques de força bruta).

Isto permite que os atacantes bloqueiem serviços aos utilizadores, bloqueando-lhes as contas.

6.4 Cross-Site Scripting (XSS)

Injeção de scripts malignos em sites.

6.5 Cross-Site Request Forgery (CSRF)

Um atacante cria links que levam para ações na página onde o utilizador já tem sessão iniciada.

6.6 Man in the Middle Attack

Intercetar comunicações entre dois sistemas.

Evita-se usando chaves públicas assinadas por uma certificate authority (CA).

6.7 Credential Storage

Passwords devem ser passadas ao servidor e só depois se faz uma hash, utilizando um salt variável.

Um salt deve ser gerado com um Cryptographically Secure Pseudo-Random Number Generator. Deve ser único para cada user. Tem que ser longo. O salt deve ser concatenado à password e, de seguida, deve ser criada uma hash. Tanto o salt como a hash devem ser guardados na base de dados.

6.8 Session Fixation

Obter ID de sessão válido, induzindo um utilizador a autenticar-se com esse ID e, posteriormente, roubar a sessão validada com o conhecimento do ID de sessão utilizado.

6.9 Session Hijacking

Ganhar controlo da sessão do utilizador roubando o ID de sessão.

6.10 Denial of Service

Tentativa de fazer uma máquina ou recurso de rede indisponível para os seus utilizadores.

7 Regular Expressions

Sequência de caracteres que formam um padrão de pesquisa.

São usadas em validação de dados, procura e substituição, e parsing.

Quando se valida, pretende-se, normalmente, que toda a string coincida com o padrão.

Quando se procura, pretende-se que uma substring coincida com o padrão.

Existem 12 caracteres especiais que têm significados especiais nas expressões regulares: \, ^, $, ., |, ?, *, +, (,), [e {. Para encontrar um destes símbolos numa string, utiliza-se uma \antes (\\, \^, etc).

Classe de caracteres, ou conjunto, coincide com apenas um de vários caracteres (exemplo: gr[ae]y: gray ou grey).

Pode-se utilizar hífens (-) para especificar intervalos numa classe de caracteres (exemplo: [0-9a-fA-F]: 0 a 9 ou a a f ou A a F).

Para negar uma classe usa-se um acento circunflexo (^) no início da classe (exemplo: [^A-Za-f]: qualquer letra exceto A a Z e a a f).

Dentro de uma classe de caracteres, os únicos caracteres especiais são], \, ^ e -.

Um ponto (.) coincide com qualquer carácter exceto mudanças de linha.

Âncoras podem ser usadas para especificar a posição da string coincidente:

- ^ coincide a posição antes do primeiro carácter da string.
- $ coincide logo após o último carácter da string.
- Podem ser usados os dois para validar uma string completa.

O metacaracter \b é uma âncora que coincide com uma posição chamada de "word boundary". Permite fazer pesquisas de palavras completas (exemplo: \b is \b: is).

Uma | permite coincidir com apenas uma de vária expressões regulares (exemplo: cat | dog).

O ? faz com que o carácter precedente seja opcional (exemplo: colou?r: color ou colour).

Uma * permite que o elemento anterior se repita 0 ou mais vezes. Um + permite que este se repita 1 ou mais vezes.

Usando { e } podemos especificar o máximo e mínimo de repetições. Exemplos:

- [0-9]{9}: repete-se 9 vezes.
- [0-9]{1, 3}: repete-se entre 1 e 3 vezes.
- [0-9]{2, }: repete-se pelo menos 2 vezes.
- [0-9]{, 3}: repete-se no máximo 3 vezes.

Para tornar as repetições lazy, adiciona-se um ? após o operador de repetição. Isto força o processador a realizar retrocessos mais vezes. Uma alternativa seria negar classes.

Colocando parte de um padrão entre () cria um grupo, que permite aplicar quantificadores e alterações a partes específicas de um padrão.

Grupos são capturados e numerados automaticamente, o que permite extrair diferentes partes de uma expressão.

Para criar um grupo que não queremos que seja capturado, começamos o grupo com **?:**

Backreferences são usadas para coincidir o mesmo texto duas vezes (\n ou $ n).

Usando ?! podemos coincidir algo que não é seguido de outra coisa.

?<= permite verificar se o texto precedente coincide (exemplo: (?<= is)land: match é land de island).

Em HTML, os elementos input têm um atributo pattern que obriga a coincidir com a expressão regular.

Em PHP, os padrões devem estar delimitados por /, # ou ~. A função preg_match() permite utilizar expressões regulares e encontrar um resultado. A função preg_match_all() encontra todos os resultados. A função preg_replace() substitui os resultados por uma string.

Em JavaScript têm que ser delimitados por /. A função test, testa se existe um resultado. A função match() executa e procura uma expressão regular numa string. A função search() retorna o índice do primeiro resultado, se existir. A função replace() permite substituir um resultado.

8 HTTP

URI: Uniform Resource Identifier

- Um identificador é um objeto que aje como referência para algo que tem identidade.
- Um URI pode ser classificado como localizador (URL), nome (URN) ou ambos.
- Componentes de um URI: esquema, autoridade, path, query, fragmento.

URN: Uniform Resource Names

- Destinam-se a servir como identificadores de recursos persistentes e independentes do local.

URL: Uniform Resource Locator

- Refere-se ao subconjunto de URI que identifica recursos através de uma representação do seu mecanismo de acesso primário, em vez de identificar o recurso por nome ou outro atributo desse recurso.
- scheme://domain:port/path?query_string#fragment_id
  - scheme refere-se ao protocolo.
  - port, query_string e fragment_id são opcionais.

Uma sessão HTTP consiste em 3 fases:

- O cliente estabelece uma conexão TCP.
- O cliente envia um pedido e espera por uma resposta.
- O servidor processa o pedido e envia a resposta, contendo um status code e os dados apropriados.

Pedido HTTP: a primeira linha contém o método seguido dos seus parâmetros. O método indica a ação a realizar.

Um método seguro não tem efeitos secundários no servidor.

- GET: usado para recuperar informação identificada pelo URI pedido.
- HEAD: idêntico ao GET, mas sem o envio do corpo da mensagem.

Um método idempotente é um método onde os efeitos secundários de vários pedidos idênticos são os mesmos efeitos secundários de um pedido.

- PUT: solicita que a entidade fechada seja armazenada no URI fornecido.
- DELETE: apaga o recurso identificado pelo URI.

Outros métodos:

- POST: solicita que o servidor aceite a entidade fechada no pedido como um novo subordinado do recurso identificado pelo URI.
- OPTIONS, TRACE, CONNECT e PATH.

Ao responder a um pedido de um cliente, o servidor envia 1 código e 3 dígitos:

- de informação (1XX).
- de sucesso (2XX).
- de redirecionamento (3XX).
- de erro do cliente (4XX).
- de erro do servidor (5XX).

Para descobrir qual método HTTP foi usado para aceder a um recurso usa-se o array $_SERVER.

- $_SERVER['REQUEST_METHOD']

9 XML

Define um conjunto de regras para codificar documentos num formato que é legível por humanos e máquinas.

É uma metalinguagem permitindo que qualquer pessoa crie a sua própria linguagem para diferentes tipos de documentos.

Um documento XML é considerado bem formado se:

- contém 1 ou mais elementos.
- tem uma e uma só raíz.
- os seus elementos se aninham adequadamente uns com os outros.

Desde o XML1.1 todos os documentos têm que começar com uma instrução que indica a sua versão, senão é considerado XML1.0. A codificação por defeito é UTF-8.

```
<?xml version="1.1" encoding="utf-8"?>
```

Secções CDATA são usadas para incluir texto que inclui blocos que, noutro caso, seriam lidos como markup. Começam com <![CDATA e terminam com]]>. Exemplo:

```
<![CDATA <warning>Not markup!</warning>]]>
```

Elementos são definidos com tag de abertura e fecho. Todos têm que ser fechados. Todos os elementos abertos dentro de outro elemento têm que ser fechados antes do pai.

Atributos são usados para associar pares nome-valor a um elemento. Só aparecem na tag inicial.

10 XPath

Linguagem para endereçar partes de um documento XML.

Modela um documento XML como uma árvore de nós.

Há diferentes tipos de nós: elementos, atributos e texto.

Os tipos de nós são usados para representar o documento como uma árvore:

- Um document node é a raíz da árvore.
- Cada nó element representa uma tag XML.
- Atributos de um elemento são representados por attribute nodes.
- Texto dentro de um elemento torna-se um text node.
- Comentários são representados como comment nodes.
- Construções XML <?...?> tornam-se processing instructions nodes.

Tipos de dados usados por expressões XPath:

- node-set: conjunto de 0 ou mais nós.
- boolean: valor true ou false.
- number: números em XPath são representados em vírgula flutuante.
- string: cadeia de caracteres.

Location Path:

- Seleciona um conjunto de elementos relativo ao content node.
- Se for precedido por /, torna-se um caminho absoluto e o context node é a raíz do documento.
- Um location path é composto por location steps, separados por /, cada um com 3 partes:
  - um eixo (axis).
  - um nó de teste (test node).
  - zero ou mais predicados.

Eixo (axis): especifica a relação de árvore entre os nós selecionados pelo location step e o context node.

Cada eixo tem um tipo principal de nó. Se um eixo pode conter elementos, o tipo de nó principal é element; caso contrário, é o tipo de nós que o eixo contém.

- Para o eixo attribute, o tipo de nó principal é attribute.
- Para o eixo namespace, o tipo de nó principal é namespace.
- Para outros eixos, o tipo de nó principal é element.

Um node test, que é um QName, é verdadeiro se e só se o tipo de nó é o tipo de nó principal e tem um nome igual ao nome especificado pelo QName. O node test * é verdadeiro para qualquer text node. O node test text() é verdadeiro para qualquer text node. O node test comment() é verdadeiro para qualquer comment node. O node test processing-instruction() é verdadeiro para qualquer processing instruction node. Um node test node() é verdadeiro para qualquer nó.

Predicados encontram-se entre [] e selecionam nós de conjunto. Um location step tem 0 ou mais predicados.

Abreviações:

- child:: -> pode ser omitido. child é o default axis.
- //e -> descendant-or-self::e
- ./e -> self::e
- ../e -> parent::e
- @e -> attribute::e

A função document.evaluate() pode ser usada para selecionar elementos usando expressões XPath. Permite selecionar elementos não selecionáveis com alguns seletores de CSS.