Guião

# Apresentação:

Olá e bem-vindos ao último curso do conjunto de cursos de *java*.

O meu nome é João Barreiras, e vou vos acompanhar ao longo deste curso avançado de *Java*. Como já devem saber, este curso tem como pré-requisito que tenham conhecimentos intermédios de *java*, e por isso aconselho-vos a terem frequentado o curso de j*ava* intermédio.

Então, e o que vamos aprender aqui?

Na primeira semana vamos aprender a criar programas que utilizem a internet, vamos falar principalmente sobre como criar programas java que acedam à internet e sobre Regular Expressions. Na segunda semana vamos falar sobre *webservices* e dos dois formatos populares de estruturação de dados, *JSON* e *XML*. O tema da 3ª semana será a programação paralela. Na semana final deste curso temos um projeto final onde vamos aplicar tudo o que aprendemos.

Como os cursos anteriores o nosso *IDE* de seleção é o *Eclipse Neon* e vamos utilizar o *Java 8*, contudo são livres de usar outras alternativas equiparáveis temos na seção de links úteis alguns tutoriais de como instalar o *Eclipse* (em *Windows*, *MAC OS X* e *Ubuntu*) e garantir que o Java está funcional, também nos links úteis têm o *link* para a nossa página do *GitHub*, onde poderão encontrar todos os exercícios e exemplos aqui apresentados, bem como outros recursos que consideramos relevantes.

Finalmente queria dar-vos os parabéns por terem chegado até este ponto, e desejo-vos boa sorte, vamos começar então?

# Semana 1

## Ligação à internet

### PowerPoint

Slide1

Bem-vindos à primeira aula do nosso curso avançado de java. Esta aula introdutória está divida em três fases. Na primeira iremos abordar o tema da ligação à internet, começando com uma revisão sobre alguns conceitos, sendo eles o TCP, o HTTP e o HTML. De seguida será apresentado um excerto de código java que permite a ligação à internet. Por fim, será proposto ao participante a realização de um exercício para aperfeiçoamento dos conhecimentos.

Slide2

Começaremos então pelo modelo TCP/IP, responsável por todas as ligações entre entidades na Internet. Este é um modelo complexo, composto por várias camadas que comunicam entre si para garantir a ligação e transferência de dados entre duas máquinas ligadas à Internet. Iremos, no entanto, abstrair-mo-nos da sua complexidade e vamos concentrar-nos na ligação peer-to-peer. Para isto vamos recorrer a uma classe java, a classe URL. Esta classe permite a criação de um objecto URL a partir de uma string, e oferece também vários métodos para a sua manipulação e utilização. Para quem esteja interessado em mais informações, disponibilizamos aqui a documentação sobre esta classe, bem como um link para mais informação sobre o funcionamento da Internet. (e outro para a origem da imagem relative ao modelo TCP/IP)

Slide3

As nossas comunicações através da Web utilizarão o HTTP, um protocolo que faz parte da camada aplicacional do modelo TCP/IP. Um simples protocolo baseado em pedidos pelo cliente, normalmente um browser web, a conteúdos alojados num servidor web, o qual responde aos pedidos enviando o conteúdo requisitado ao cliente. Há também, como se vê na imagem, a possibilidade de o servidor enviar cookies para o cliente, os quais são depois enviados para o servidor juntamente com futuros pedidos, permitindo aprofundar a comunicação entre servidor e cliente. No java existem três classes principais que utilizam este protocolo para a ligação à Internet: a classe URL/URL connection, na qual nos iremos focar nesta aula, a classe HttpComponents, pertencente à biblioteca Apache, e a classe Socket.

Slide4

Além do HTTP, responsável pela comunicação de dados, é também do nosso interesse conhecer o HTML, a linguagem utilizada para formatar páginas web. Esta é uma linguagem que permite a representação estruturada de conteúdos texto e multimedia, focada no uso de tags. Uma tag consiste num texto padrão, delimitado pelos sinais de menor que e menor que, e que é iniciada, aparecendo depois desta o conteúdo ao qual queremos que esta se aplique, e finalmente a tag é fechada escrevendo novamente a tag com uma barra no início do seu texto, marcando o fim do conteúdo afectado pela tag. Este conteúdo pode ser texto, links para imagens ou para outros websites, etc. Ao trabalhar com a internet é essencial perceber esta linguagem e conhecer as principais tags, e as nossas aulas não serão excepção.

Slide5

Vamos agora passar ao código java. Este excerto de código, encapsulado num try/catch para evitar erros em tempo de execução que podem ocorrer ao trabalhar num ambiente web, começa num primeiro momento por criar um novo objeto da classe URL, o qual será inicializado com a hiperligação para a página do github referente a este exercício. Uma String onde armazenaremos a resposta do servidor web também é criada neste momento. De seguida iniciamos uma ligação utilizando o URL anterior, que irá resultar numa resposta do servidor web contendo a página web pedida. Os dados devolvidos pelo servidor são então recebidos por uma InputStream, e armazenados num buffer. Agora é apenas necessário guardar o texto, que neste momento está armazenado no buffer, na String criada para receber a informação da página. Um ciclo while permite receber todo o texto do buffer, que é por fim escrito na consola.

Slide6

O exercício anterior permite-nos receber, armazenar e apresentar o conteúdo de uma página, mas não é uma solução ideal. Como se pode ver no código html no topo, que corresponde ao resultado do código anterior, é difícil ler a informação contida na página, já que o HTML não foi criado para ser utilizado para armazenar dados, mas para os estruturar para apresentação em páginas web. Assim, iremos na próxima semana abordar o XML e o JSON, duas alternativas interessantes para este tipo de transferência de dados através da internet.

Slide7

Para terminar, temos então uma proposta de exercício, para quem quiser praticar o que aprendemos neste vídeo. Este exercício simples não é muito diferente do nosso exemplo, mas desta vez o objetivo é especificamente guardar (ou as hiperligações das imagens?) as imagens presentes na página indicada. Se tiver dificuldades e não conseguir resolver o exercício não se preocupe, que o nosso próximo vídeo começará com a resolução deste exercício. Chegámos assim ao fim da nossa primeira aula, obrigado pela vossa presença e esperamos ver-vos no próximo vídeo!

### Exercício

#### Enunciado

Escrever um programa java que aceda ao url: <https://java-mooc.github.io/Advanced-Java/ex2.html> e faça o download de todas as imagens.

## Regular Expressions

### PowerPoint

### PowerPoint (website)

Texto exemplo:

*King Arthur, also called Arthur or Arthur Pendragon, legendary British king who appears in a cycle of medieval romances (known as the Matter of Britain) as the sovereign of a knightly fellowship of the Round Table. It is not certain how these legends originated or whether the figure of Arthur was based on a historical person. The legend possibly originated either in Wales or in those parts of northern Britain inhabited by Brythonic-speaking Celts. (For a fuller treatment of the stories about King Arthur, see also Arthurian legend.)*

*Assumptions that a historical Arthur led Welsh resistance to the West Saxon advance from the middle Thames are based on a conflation of two early writers, the religious polemicist Gildas and the historian Nennius, and on the Annales Cambriae of the late 10th century. The 9th-century Historia Brittonum, traditionally attributed to Nennius, records 12 battles fought by Arthur against the Saxons, culminating in a victory at Mons Badonicus. The Arthurian section of this work, however, is from an undetermined source, possibly a poetic text. The Annales Cambriae also mention Arthur’s victory at Mons Badonicus (516) and record the Battle of Camlann (537), “in which Arthur and Medraut fell.” Gildas’s De excidio et conquestu Britanniae (mid-6th century) implies that Mons Badonicus was fought in about 500 but does not connect it with Arthur.*

*Early Welsh literature quickly made Arthur into a king of wonders and marvels. The 12th-century prose romance Kulhwch and Olwen associated him with other heroes, and this conception of a heroic band with Arthur at its head doubtless led to the idea of Arthur’s court*

1. Só texto

Ex:

* Arthur
* See
* Banana

1. ^ (ínicio de uma frase)
   * ^King
   * ^Early
   * ^Kid
2. $ (fim de uma frase)
   * Arthur.$
   * Shoe.$
3. . (um carater único)
   * .
   * Ar.
   * …
   * Shoe.
4. \s (espaços em branco)
   * \sArthur\s
   * \stomorrow
5. \S (tudo menos espaços em branco)
   * \Srthur
   * \S+
   * \S\S
6. \* (0 ou mais vezes c/ ? greedy; s/ ? non-greedy)
   * (Arthur)\*
   * (literalmente o que quiseres)\*
7. + (1 ou mais vezes c/ ? greedy; s/ ? non-greedy)
   * (Arthur)+
   * .n+
   * (literalmente o que quiseres)+
8. [] (quaisquer elementos dentro do conjunto)
   * [aeiou]+
   * [84]
9. [^] (quaisquer elementos que não estejam dentro do conjunto)
   * [84]+
   * [aeiou]
10. [x-y] (de x a y)
    * [a-z]+
    * [a-zA-Z]
    * [0-9A-Z]

### Exercício

## Quizz Final

# Semana 2

JSON/XML

### PowerPoint

## Webservices

### PowerPoint

### Exercício

## Quizz Final

# Semana 3

## Paralelismo de tarefas

### PowerPoint

### Exercício

## Threads

### PowerPoint

### Exercício

## Sincronização de dados e pipelining

### PowerPoint

### Exercício

## Quizz Final

# Semana 4 (Projeto Final)

## Enunciado

## Objetivos