**Primeiros Passos com Angular**

**Qual a responsabilidade do arquivo package.json?**

**R:** Definir dependências necessárias para aplicação.

**O que é um Componente?**

**R:** Define a área de interface do usuário (UI), que permite a reutilização desse conjunto de funcionalidades.

**As propriedades de básicas de um Componente são:**

**R:** Classe de Componente, HTML Template e um estilo específico.

**Qual a função do comando ng serve?**

**R:** Iniciar a aplicação.

**O que são componentes aninhados?**

**R:** São componentes que são adicionados em outro componente, passando ou não informações para este.

**Qual a responsabilidade do arquivo angular.json?**

**R:** Definir propriedades genéricas na aplicação como folhas de estilos, javascript, arquivo responsável por iniciar a aplicação main.ts, etc .

**O que é um Decorator?**

**R:** Define uma classe como um componente, module, pipe, service, e seus metadados que determina como o mesmo deverá ser processado.

**A anotação @Input é utilizada para:**

**R:** Tornar uma propriedade de um componente elegível para receber um valor de um outro componente.

**A anotação @Component é utilizada para:**

**R:** Tornar uma classe elegível para ser um Componente.

**Qual a função do comando ng new?**

**R:** Criar um projeto.

--------------------------------------------------------------------------------------------

**Lidando com Vários Componentes**

**Qual o conceito de one-way data bind?**

**R:** Liga uma propriedade do componente em uma única direção (view).

**A anotação @Pipe é utilizada para:**

**R:** Tornar uma classe elegível para ser um pipe.

**Uma classe no Angular, poderá receber a injeção de dependência de um outro objeto via:**

**R:** Através do construtor.

**A anotação @Injectable é utilizada para:**

**R:** Torna uma classe elegível para injeção de dependência.

**Qual o conceito de two-way data bind?**

**R:** Liga uma propriedade do componente em duas direções (view/change).

**Para acessar o método responsável pela transformação de um pipe, é necessário realizar a implementação da interface:**

**R:** PipeTransform.

**O que são Pipes?**

**R:** Responsável por, receber, transformar e exibir uma informação para os usuários.

**Qual a função do código routerLink?**

**R:** Navegar entre views através de um link de rota.

**Qual a função do RouterModule?**

**R:** Habilita a navegação de uma view para outra pelo usuário.

**O que é injeção de Dependência?**

**R:** São objetos de serviço responsável por executar sua função, cuja sua instância é compartilhada para quem os chama.

--------------------------------------------------------------------------------------------

**Segregando Responsabilidades**

**No decorator da anotação @NgModule, a propriedade imports é utilizada para:**

**R:** Declarar outros módulos necessários para o comportamento interno do módulo em questão.

**No decorator da anotação @NgModule, a propriedade exports é utilizada para:**

**R:** Exportar componentes/pipes internos do módulo para serem utilizados por outro módulo.

**A anotação @NgModule é utilizada para:**

**R:** Tornar uma classe elegível para ser um módulo.

**Caso um componente seja declarado no módulo raiz (app.module) e em outro módulo, e este outro módulo seja importado pelo módulo raiz, qual das opções define esse evento?**

**R:** A aplicação será iniciada, mas não será possível utilizá-la, pois não é possível ter dois componentes declarados em módulos diferentes.

**Para realizar uma requisição HTTP, devemos utilizar a injeção de dependência da seguinte classe:**

**R:** HttpClient.

**Qual a função do HttpClientModule?**

**R:** Provê todas as dependências necessárias para realizar uma requisição HTTP.

**No decorator da anotação @NgModule, a propriedade declarations é utilizada para:**

**R:** Declarar os componentes/pipes internos do módulo em questão.

**Qual o conceito de módulos em Angular?**

**R:** São contêineres, ou limitadores de contexto, para um bloco de código coeso de um área de domínio da aplicação.

**Qual o conceito de Observable?**

**R:** Trata respostas de eventos assíncronos utilizando o padrão Publisher.

**Caso um Observable não seja subscrito, qual dos comportamentos abaixo estará correto:**

**R:** O evento do Observable não será executado.