**Exercícios EBAC**

1.O que são *Generics* e que problemas eles resolvem?

O *Generecis* <T> é um tipo que pode receber valores de qualquer outro tipo.

Por conta disso, ele permite que escrevamos funções de tipos flexíveis e reutilizáveis, evitando a repetição de código, e centralizando a funcionalidade em um único local, para qualquer tipo.

2.De um exemplo de uso de *Generecis*.

Podemos utilizar o *Generics* em uma função que realiza o *decode* de um tipo genérico T, de um JSON ou de um *request* na API.

Dessa forma, se surgisse a necessidade de realizar a decodificação de vários tipos diferentes ao longo do programa, não precisaríamos criar uma função que realizasse o *decode* de cada tipo separadamente, tendo em vista que o processo é o mesmo, o que mudaria seria apenas o tipo.

Descrição gerada automaticamente


3.Quais são os cuidados que deve tomar, ao usar políticas de caches?

As politicas de cache facilitam o download e uploads de arquivos da API, e ajudam os desenvolvedores a identificar quando há necessidade de realizar uma chamada para API ou não.

É necessário se manter atento para não selecionar uma política de cache que sempre realiza uma requisição para API, pois isso pode ser custoso para a organização, além de consumir a internet do usuário e cortar o acesso quando este não tiver acesso a internet.

Outro ponto, é manter uma política de cache que sempre pega os arquivos do cache do celular, pois se o cache estiver vazio o aplicativo vai crachar.

4.De um exemplo de uso de Clousure com escaping e sem escaping, observando o uso do weak self.

O weak self deve ser utilizado somente quando a clousure “escapa” do contexto, quando ela tem um @escaping.

Exemplo de uma clousure com escaping:

func loadCharacter(usging @escapging clousure: ()->Character){

//process

}

Exemplo de uma clousure sem escaping:

func loadCharacter(usging clousure: ()->Character){

//process

}