DICAS & TRUQUES

-4-
EDIÇÃO

25/ABR/2020

INTRODUÇÃO

Esta 4ª edição é direcionada a todos os profissionais, entusiastas em tecnologia e empresas que pretendem saber mais sobre as ***PWA's*** (Progressive Web apps) para o desenvolvimento **offline**.

PARTILHADA POR:

EDILSON COELHO

CONSULTOR SÊNIOR IT

923242277

CARLOS SPRANGER

CONSULTOR SÊNIOR IT

946145301

[FAÇA PARTE DA COMUNIDADE NO LINKEDIN]

Falar de **PWA** e nos esquecer-mos da evolução da web é um erro caro para os leitores, **PWA** faz parte do processo de evolução da **Web**, ou seja, os vários estágios até ao ponto que nós consideramos ser o mais alto “marco” as **PWA’s**.

SITES ESTÁTICOS **SITES DINÂMICOS** **AJAX** **DESIGN RESPONSIVE** **PWA’s**

02

O que é **PWA** (Progressive web apps)? Nós entendemos que **PWA** é uma evolução híbrida que proporciona as experiências já conhecidas em páginas comuns da web e aplicações móveis.

As **PWA's**, no nosso ponto de vista surgem como alternativa para fazer frente ao desenvolvimento mobile usando tecnologias híbridas ou nativas, quando pensamos em construir uma app nos dias de hoje, na nossa perspectiva, faz sentido colocarmos todas as cartas na mesa sobre o grau de complexidade que a app irá exigir, sabem por quê?

As **PWA's** proporcionam a mesma experiência de uma app, elas podem ser instaladas no teu desktop ou no seu telefone dando a ilusão de uma aplicação mobile graças a peça fundamental de uma **PWA**, os chamados **SW** (Service workers).

04

Referenciamos o service worker como peça fundamental, realmente são a mágia de uma **PWA**. O **SW** é um script em que os navegadores executam em segundo plano sem estar associado ou vinculado a nenhuma página de uma determinada aplicação web.

- **a.** O **SW** já inclui um conjunto de recursos nomeadamente:

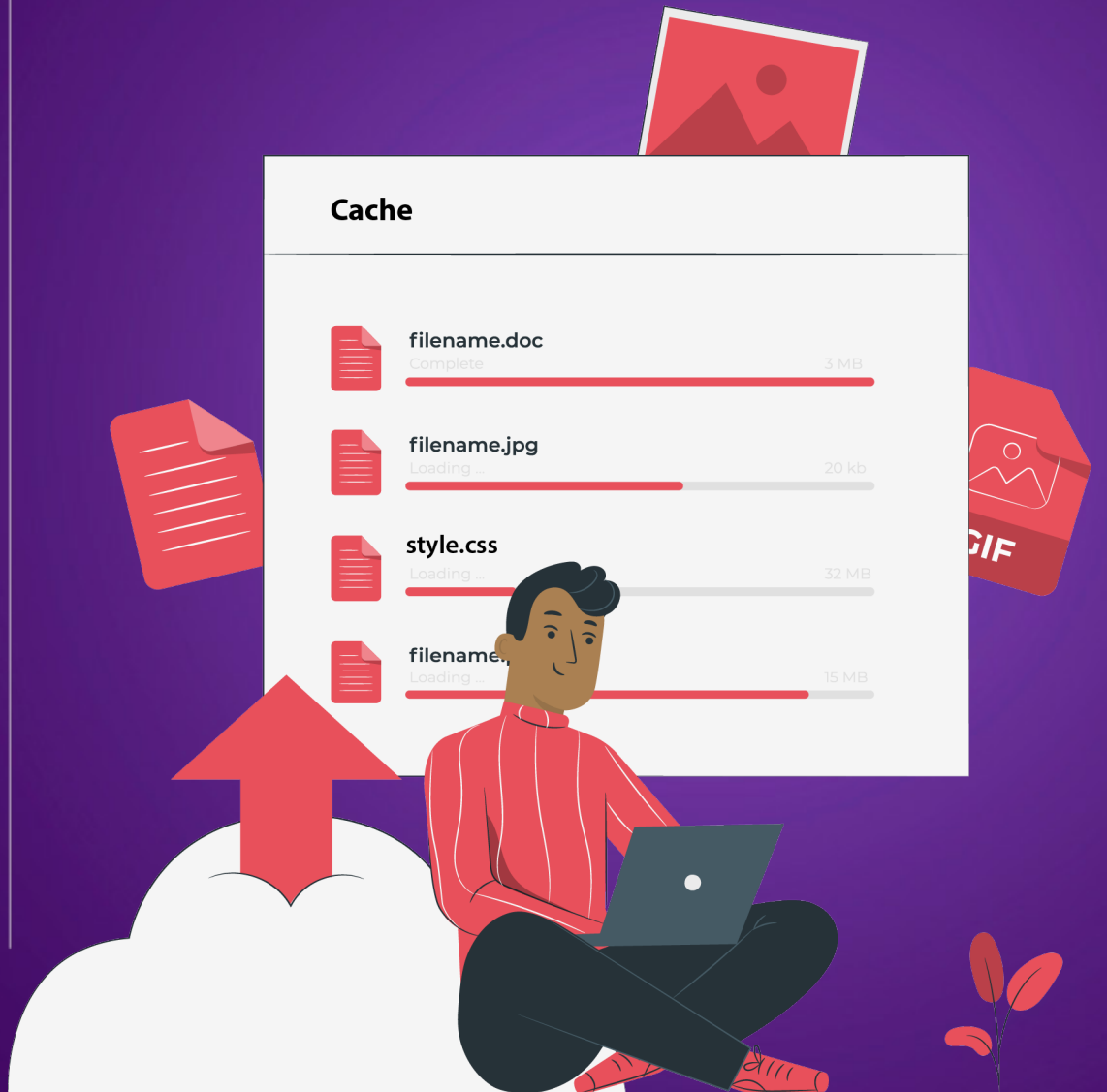
  i. As notificações de PUSH

  ii. Sincronização em segundo plano

05

As **PWA's** nos permitem definir estratégias de cacheamento dos nossos recursos, como arquivos de imagens, ficheiros ***.js**, ficheiros ***.css** e resultados de requisições **http** a serviços da web. Será que nos dias de hoje onde a capacidade de resposta é um diferencial, faz sentido solicitar ao servidor para carregar sempre arquivos estáticos? O cache no cliente evita inúmeras requisições no servidor web e aproveita os recursos de uma **PWA** para trabalhar **offline** sempre que necessário.

Você pode colocar url's de endpoints em cache, evitando requisições desnecessárias a recursos externos, e sempre que a conexão falhar, você pode ir buscar os dados que se encontram em cache no navegador "*Melhorias na experiência do utilizador*".



06

Por força da própria mudança dos utilizadores e exigências, proporcionar a melhor experiência ao consumidor final fazem toda diferença; isso para dizer que podem existir cenários de perda de conexão no momento em que o utilizador encontra-se perante ao preenchimento de um formulário para ser submetido.

Será que por falta de conexão iriamos atrasar o seu trabalho ou forçar o mesmo a preencher os dados novamente quando restabelecer a conexão? Difícil decisão, mas se pensarmos na abordagem **offline** teríamos outras perspectivas:

- **a.** Realizar o armazenamento local em uma base de dados como **IndexDB** que suporta grandes volumes de dados e garante performance para armazenar os dados do formulário em modo **offline**;

- **b.** Sempre que a conexão restabelecer, você deve iniciar sempre o processo de sincronização com o servidor para que seja registado os dados do formulário na base de dados final (Mysql, Oracle, SQL Server, DB2, etc);

- **c.** Sabendo que o **SW** (Service workers) é executado em segundo plano, teremos sempre a garantia de que podemos executar scripts em background e executar determinadas tarefas mesmo tendo o nosso site, portal fechado no navegador “browser”;

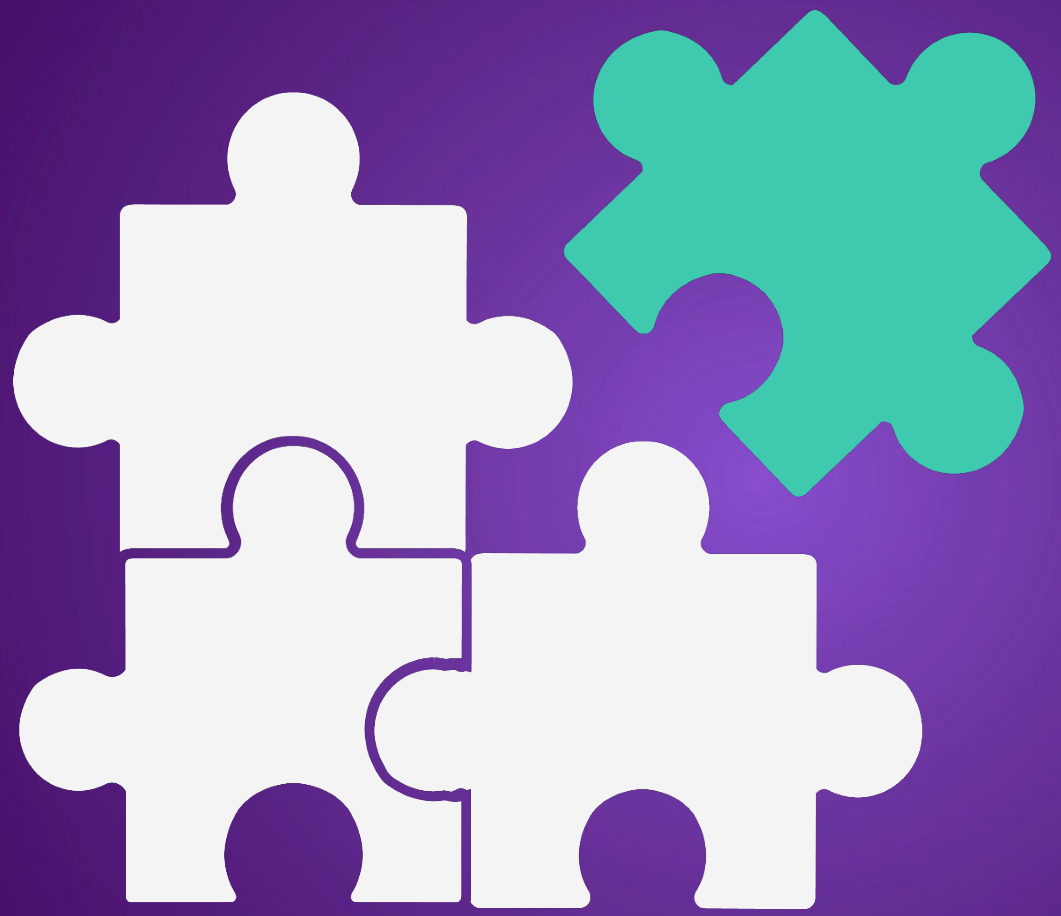
Com as **PWA's** é possível definir várias estratégias de **cache** com a finalidade de proporcionar uma melhor experiência para os utilizadores, mesmo estado **offline** "sem conectividade" nomeadamente:

- **a.** Cacheamento de todos arquivos estáticos no cliente:
  - **i.** Arquivos CSS
  - **ii.** Arquivos JS
  - **iii.** Outros __ eot|svg|cur|jpg|png|webp |gif|otf|ttf|wof |woff2|ani

- **b.** Cacheamento de url's:
  - **i.** Endereços de arquivos externos (fontes, imagens, css, js, etc)
  - **ii.** Endereços de API's para acesso aos dados:

1. Neste cenário, você pode garantir que determinadas informações possam ficar em cache de formas a evitar que enumeras solicitações sejam feitas no servidor "network" e o próprio carregamento **offline**.

Atenção, não significa que **PWA** é a solução para tudo que você necessita para desenvolver para a web, achamos ser mais uma alternativa para resolver constrangimentos que podemos encontrar em determinados cenários.



09

Não fique preso no legado, seja flexível a mudanças, acompanhe o que há de novo no mercado sempre que possível, não seja mente rígida e defensor de que as tecnologias que você domina “zona de conforto” são as melhores ou que atendem todas as necessidades. Tempo para aprender **PWA**, nunca é tarde as vezes é cedo de mais.

“Se puderes, ajuda os outros; se não o puderes fazer, ao menos não lhes faças mal”.

DALAI LAMA