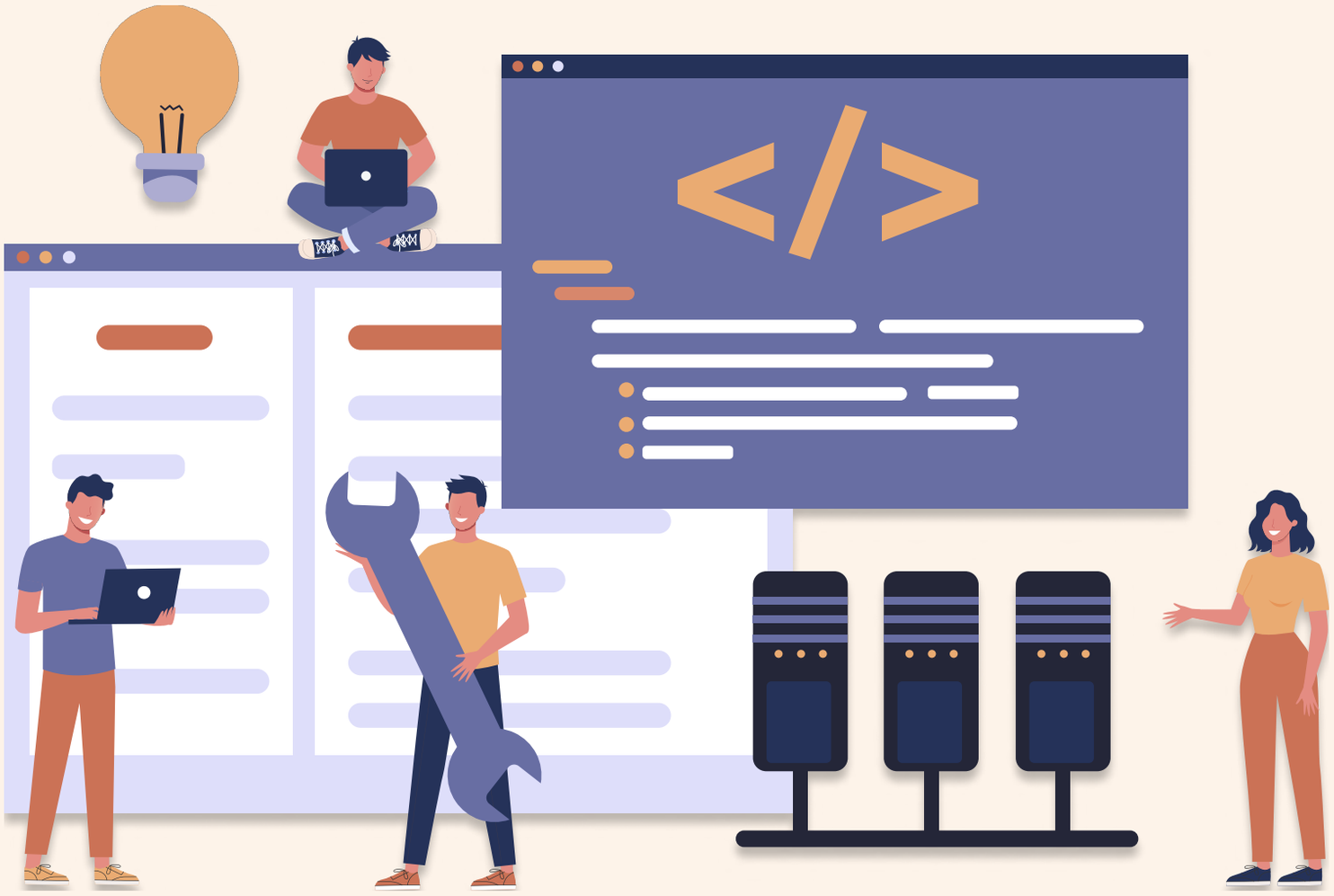
An Extensive Guide to Progressive Web Apps



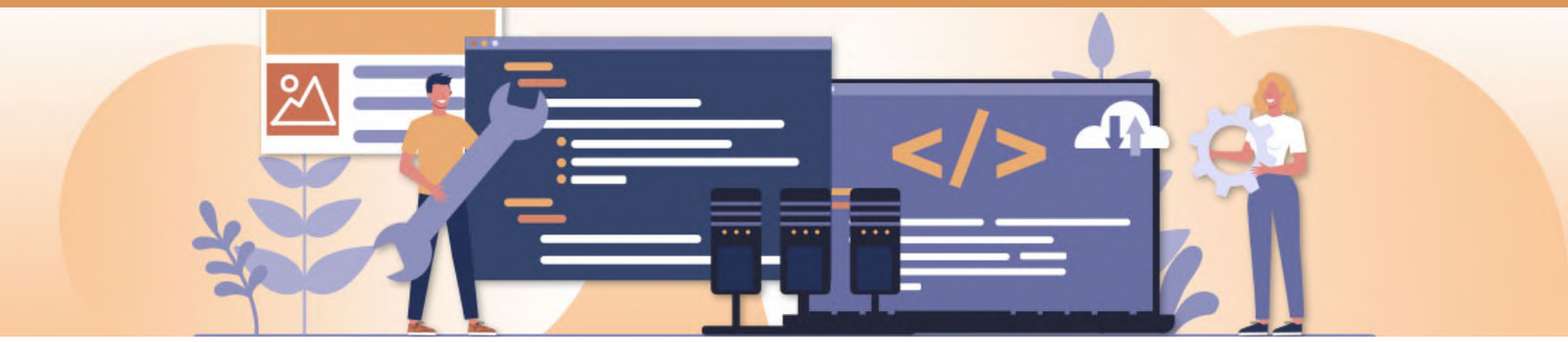
For the past several years, the way we access the internet has significantly changed. Now, mobile phones generate 51.53% of global website traffic. The world is becoming more mobile and we get more accustomed to consuming internet data on the go.

This change brings mobile and web technologies to a new level. New cutting-edge solutions appear facilitating business-customer interaction and enhancing end-user experience. One of these latest trends is progressive web apps (PWA). These apps represent a perfect blend of web and native apps. It means that they are easier to access and navigate while cheaper for companies to develop.

In this guide, we'll uncover:

- What are PWAs?
- How do PWAs work?
- Benefits of progressive web apps over mobile apps.
- Common PWA development tools.

What Are Progressive Web Apps?



In short, a progressive web app (PWA) is a type of a web app that includes a combination of various technologies, design concepts, and Web APIs that provide user experience similar to native mobile apps. It means that PWAs use web technologies in their core while performing the same functionalities and providing native-like experience.
The idea of blending web and native apps in PWAs was first presented by Google in 2015 and since then their popularity is constantly growing. Many companies state that PWAs helped them to double user activity and increase their conversion rates.

How Do PWAs Work?

The most striking feature of all PWAs is that they run in a web browser. Web browsers work as virtual machines for these apps, they run PWAs in their environments. While native apps require Android or iOS operating systems to launch, PWAs need only a browser.
Most of the web browsers support PWAs. However, some are just working on integrating the technology (Internet Explorer or Safari), while others have already implemented it (Google Chrome, Mozilla Firefox, Edge, and Brave).

PWA Architecture includes:

Service Worker is an abstraction layer that links the backend and frontend in a browser. It represents a javascript file that connects to the HTML page code. Service Worker handles all the user requests and has cache and database access for storing data. It means that even if a user goes offline or has a low internet connection, a PWA will still work.

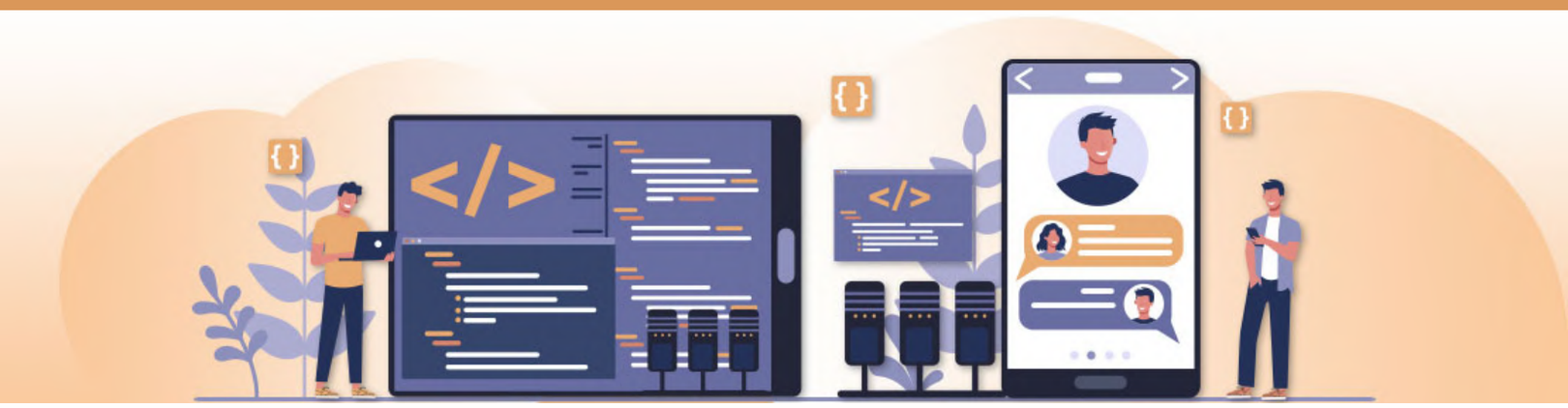
Web App Manifest is the JSON file that determines the main parameters of a PWA. For example, it can define an app's name in a browser, what app icon to use, how PWA will be displayed, and many other features. App Manifest installs PWA on a mobile screen as a standalone application.

HTTPS protocol ensures that the data a user sends via PWAs is secure. HTTPS encrypts the data with the SSL protocol. It means that software engineers can use PWA to create banking or finance apps where personal data encryption is vital.

Application Shell is the thing that makes PWAs look like native apps. An app shell represents a skeleton of the Graphical User Interface (GUI) for PWAs. It consists of the minimal HTML, CSS, and JavaScript and allows instantly loading the basic elements of an app and insert the loaded data into it.

Push Notification is a small popping up message on a website. It usually allows users to install an app on their mobile screen and subscribe for new updates and notifications. Afterwards subscription notifications with the latest news and updates will occasionally appear on a user's device even without visiting the website.

Benefits of PWAs over Mobile Apps



For mobile app development, software developers need to learn various programming languages depending on the platform. For example, they need to learn Objective-C for iOS and Java for Android, while PWAs use only HTML, CSS, and JavaScript and can be run on any device via a browser. PWAs have the word 'progressive' in their name which means that they are aimed at enhancing performance in comparison to other apps.

The main benchmarks are reliability, speed, and engagement. PWAs are:

- **reliable** as their performance doesn't depend on the internet connection quality and they can work successfully even offline;
- **fast** as they exchange data with the web fast and smoothly and their user interfaces hardly ever lag;
- **engaging** as they provide a native-like app user experience.

Let's compare PWAs to other mobile apps in more detail.

PWAs vs Native Apps

Native apps are built for a particular platform or a gadget. They have access to all the device features such as a microphone, camera, GPS, and others and have better integration with a device operating system and hardware. Native apps have strong benefits, however, there are some features where they lose to PWAs:

1. PWAs have lower development costs as they are cheaper to maintain and update than native apps. Moreover, you don't need to search for unique specialists to build special features on a particular programming language.

2. It is faster to develop PWAs than native apps as they fit in various mobile operating systems.

3. PWAs take less storage space in comparison to Native Apps. Native Apps require full download and installation on a device, while PWA's are stored on web servers.

4. It is easier to access and distribute PWAs than native apps. To get a native app, users need to go to the app stores while PWAs are placed at users' fingerprints on the same webpage they're visiting and can be shared with a URL link. Moreover, there is no need to fulfill certain pass requirements and meet the standards set by app stores.

5. Native apps are not listed in search engines hence they lose the battle for popularity. PWAs, in turn, can be placed in search engine results, therefore they can compete for better ranking.

6. Although PWAs have a secure HTTPS connection, native apps have more ways to establish security measures, like multi-factor authentication and passing security requirements in app stores.

7. Native apps are more powerful and faster than PWAs. Above all, native apps can perform many functions that PWAs can't, for example, proximity sensors, mobile payment, smart lock, or interaction with other apps e.g. calendar, making calls, and others.

PWAs vs Hybrid Apps

Hybrid apps are the apps that combine features of a native and web application. This technology was invented as a response to single platform development in Native Apps.

1. Hybrid Apps' responsiveness depends on a website's user interface components.

2. Hybrid Apps are Appstore dependent while PWA apps aren't.

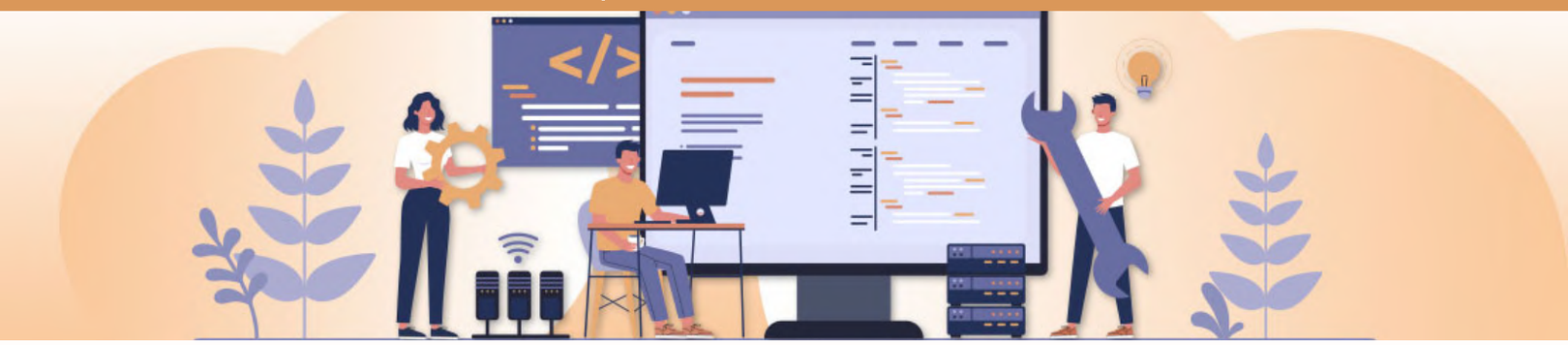
3. PWAs are easily customized in comparison to Hybrid apps.

4. Development of Hybrid Apps requires deep knowledge in Native App development

5. PWAs are cheaper to develop in comparison to Hybrid Apps as they have a single code base and don't require versatile development skills.

6. PWAs significantly exceed Hybrid Apps by performance as they have a simpler code structure and they are fast even with slow internet.

What Are Common PWA Development Tools?



PWA developers usually utilize the same frameworks and tools for the development of common web apps. Here is a list of the most popular PWA development tools:

1. **React.JS** is one of the best web development frameworks, therefore, it suits well for building PWAs. It supports only View mode from a standard Model-View-Controller development model and allows embedding HTML into JavaScript files with the virtual DOM that renders web-pages on a server-side. This framework is one of the most popular solutions for web development as it's easy to learn and quite flexible to use.

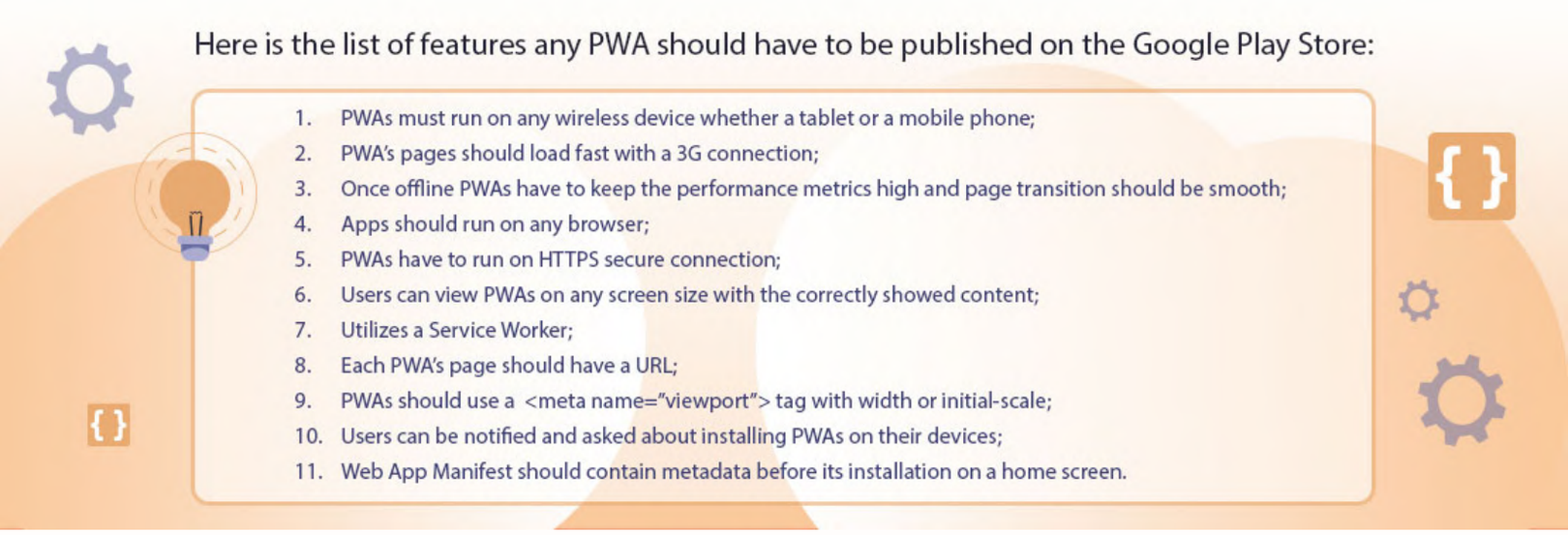
2. **Knockout** is a free JavaScript library that helps to build lightweight PWAs. This small (13 KB), yet multifunctional library allows building dynamically changed interfaces.
Knockout provides software developers with templates that facilitate the creation of complex apps by minimizing the duplication of DOM elements and embedding them in HTML easier. Also, Knockout uses HTML5 data to bind HTML elements to data objects in JavaScript and doesn't require any JSX as in the case of React.

This JavaScript library is a real catch for web engineers as it significantly facilitates PWA development.

3.Lighthouse is an automated tool for enhancing web page quality developed by Google. Software developers can use Lighthouse as a plugin in Chrome to run tests on how a PWA performs offline, loads pages, provides authentication, and much more. This open-source audit performance tool highlights the possible program flows and helps web developers build robust and effective solutions.

4.Webpack is a tool used to compile JavaScript modules into a browser. Many web developers use Webpack as it allows effectively building the front-end. Webpack helps to build a Service Worker and a Manifest which are the essential parts of any PWA. This powerful tool allows building stable and high-quality PWA solutions.

The progressive web app technology is evolving quickly. To keep the standards of PWA quality and development high, Google has recently released a Progressive Web Apps checklist that includes the most essential features every PWA should have.



Conclusion

Progressive Web Apps have already proved for many businesses to be a technology with considerable benefits. They allow companies to better reach their audiences, retain their customers, and increase sales.

Moreover, PWAs are much cheaper to develop and maintain in comparison to native and hybrid apps. Therefore, we expect that in the near future more businesses will turn their attention to PWAs to reap commercial benefits and stay closer to their customers.

Feel free to contact us and ask any questions: info@scand.com