W3C MiniApps Overview

Qing An
Co-chair of MiniApps WG/CG
Alibaba Group
2023-06-20

阿里巴巴集团标准化部
ALIBABA STANDARDIZATION DEPARTMENT

Outline

1. MiniApps Ecosystem
2. W3C MiniApps Introduction

MiniApp can be a new carrier of W3C standards

MiniApp can work as a new carrier

to implement and enhance Web standards, lead the Web to its full potential

e.g.

HTML5

Web Payments

WebRTC

Web of Things

Accessibility

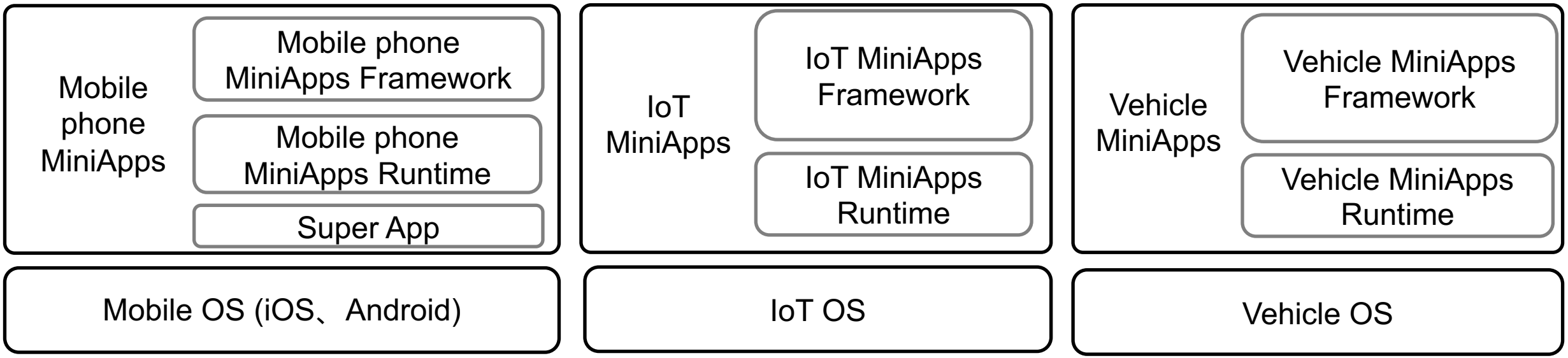
Digital Publishing

CSS

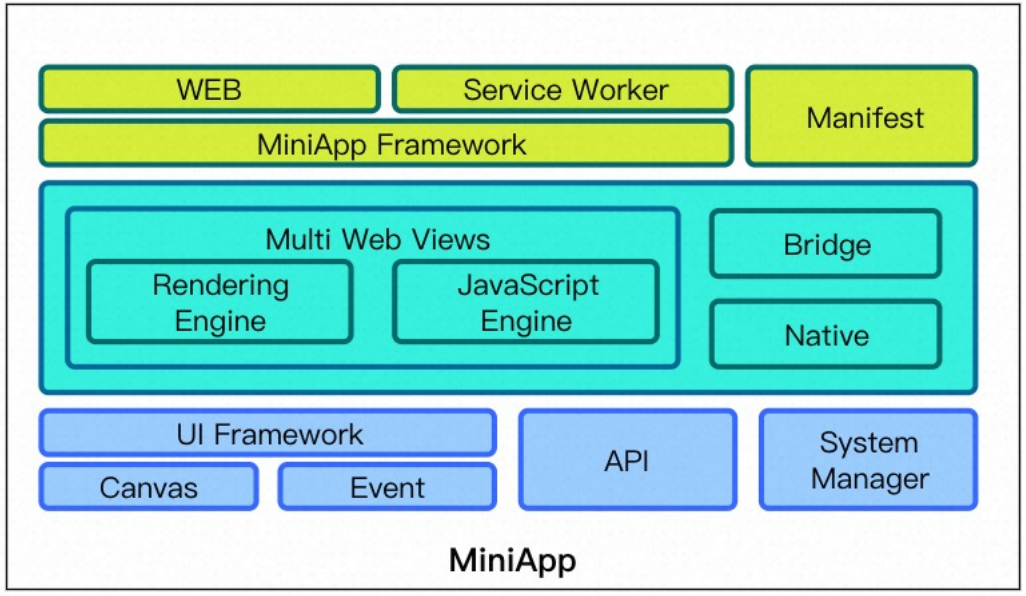
Automotive

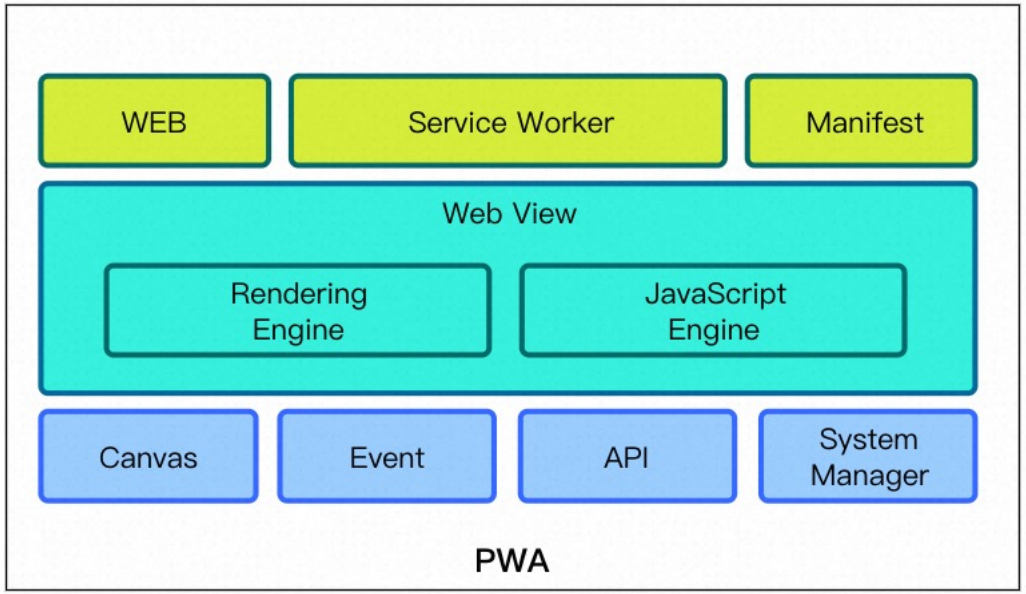
MiniApps

MiniApps ecosystems



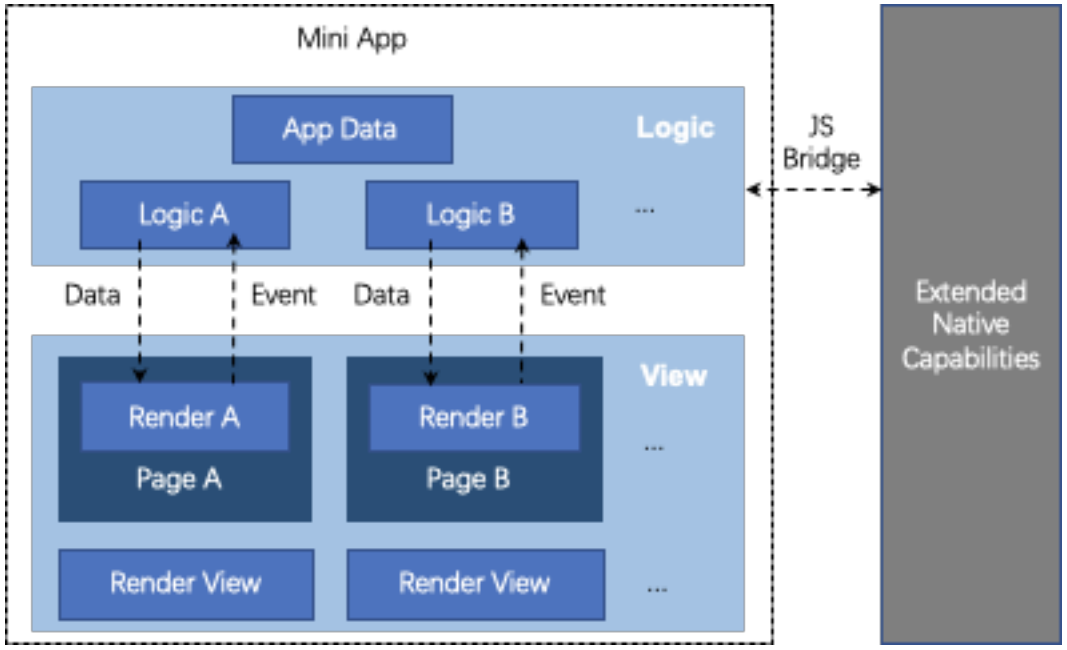
MiniApps and PWA

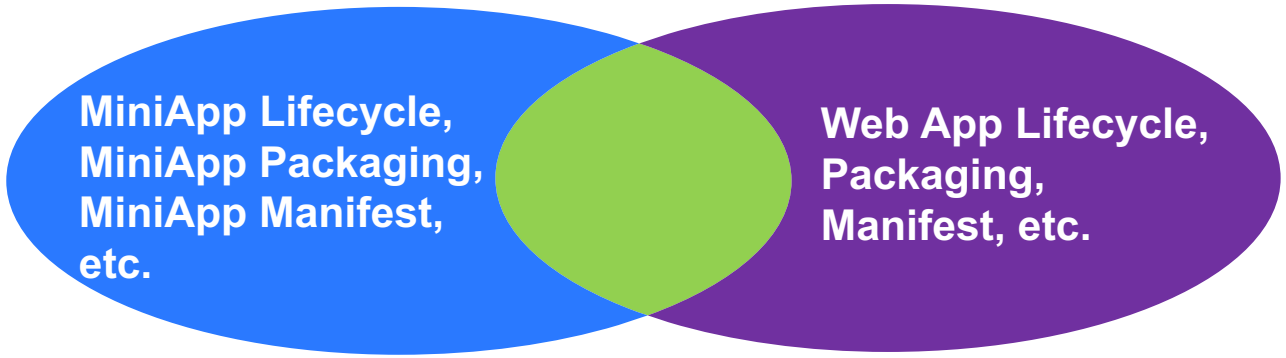




Feature	Progressive Web App	MiniApp
Source code	Standard markup languages (HTML), stylesheets (CSS), and scripts (JavaScript).	Non-standard dialects of HTML, CSS and JavaScript
Deployed Format	Web resources (mainly: HTML, CSS, JavaScript code, and WebAssembly modules)	HTML, CSS, JavaScript, and other resources packed in a ZIP container.
Packaging	No. Resources linked on the Web.	Yes. Different package formats per vendor.
Needs to host files on Web server	Yes	No
Installation-free usage	Yes, running in the browser.	Running in a super app or on the OS.
Installation with standalone icon	From the browser or app marketplace (optional)	No
Services	Access to Web APIs	Access to non-standard Web APIs, including some system native APIs

W3C MiniApp Architecture





W3C MiniApps WG

W3C MiniApps Working Group: https://www.w3.org/2021/01/miniapps-wg-charter.html

- In progress：
  1. MiniApp Manifest
  2. MiniApp Packaging
  3. MiniApp Lifecycle
  4. MiniApp Addressing
  5. Widget Requirement

MiniApps Working Group Charter

The **mission** of the MiniApps Working Group is to produce specifications that facilitate the development of interoperable and robust MiniApps.

Join the MiniApps Working Group.

End date	2023-01-20
Chairs	Anqi Li (Alibaba), Yongjing Zhang (Huawei), Ming Zu (Baidu)
Team Contact	Fuqiao Xue (0.2 FTE)
Usual Meeting Schedule	Teleconferences: topic-specific calls may be held Face-to-face: at least 1 per year

MiniApp Manifest

Manifest: https://w3c.github.io/miniapp-manifest/

Member	Type	Required	Description
app_id	string	Yes	MiniApp identifier
color_scheme	string	No	MiniApp color scheme
description	string	No	MiniApp description
device_type	list	No	Supporting devices
dir	string	No	Direction of texts
icons	image resource list	Yes	MiniApp icons
lang	string	No	MiniApp primary language
name	string	Yes	MiniApp name
pages	list	Yes	Page routing information
platform_version	platform version resource	Yes	Platform version supported
req_permissions	permission resource list	No	Required permissions
short_name	string	No	MiniApp short name
version	version resource	Yes	MiniApp version
widgets	widget resource list	No	MiniApp widgets
window	window resource	No	Window style

MiniApp Packaging

Packaging: https://w3c.github.io/miniapp-packaging/

To ***process a MiniApp package***, given URL *miniapp_uri*, perform the following steps:

1. Let *miniapp_zip_file* be the result of retrieving a MiniApp ZIP container with *miniapp_uri*.
2. Verify a MiniApp ZIP container with *miniapp_zip_file*.
3. Let *miniapp_package* be the result of unzipping the *miniapp_zip_file*.
4. If *miniapp_package* is an unzip exception, then return failure.
5. Let ordered map *manifest* be the result of processing the MiniApp manifest with *miniapp_package*.
6. Let *start_page* be the result of preparing the platform runtime with *miniapp_uri* and *manifest*.
7. Let string *locale* be the result of extracting the locale, passing *manifest*.
8. Launch MiniApp passing *miniapp_package*, *manifest*, *start_page*, and *locale*.

EXAMPLE 1: File system structure

```
/
|___manifest.json
|___app.js
|___app.css
|___pages/
|       |___page1.js
|       |___page1.html
|       |___page1.css
|___common/
|       |___componentA.js
|       |___componentA.html
|       |___componentA.css
|       |___example.png
|___i18n/
        |___zh-Hans.json
        |___en-US.json
```

MiniApp Lifecycle

Lifecycle: https://w3c.github.io/miniapp-lifecycle/

2.4 MiniApp Global Application Lifecycle interface

WebIDL

```
[Exposed=Window]
interface Global {
 readonly attribute GlobalState globalState;
 readonly attribute InputObject inputObject;
 readonly attribute LifecycleError lifecycleError;
 attribute EventHandler ongloballaunched;
 attribute EventHandler onglobalshown;
 attribute EventHandler onglobalhidden;
 attribute EventHandler onglobalerror;
 attribute EventHandler onglobalunloaded;
};
```

3.4 MiniApp Page Lifecycle interface

WebIDL

```
[Exposed=Window]
interface Page {
    readonly attribute PageState pageState;
    readonly attribute PageInputObject pageInputObject;
    attribute EventHandler onpageloaded;
    attribute EventHandler onpageready;
    attribute EventHandler onpageshown;
    attribute EventHandler onpagehidden;
    attribute EventHandler onpageunloaded;
};
```

MiniApp Addressing

Addressing: https://w3c.github.io/miniapp-addressing/

The MiniApp URI syntax is defined using [ABNF], using `host`, `path-abempty`, `query`, `fragment`, and `unreserved` from [RFC3986].

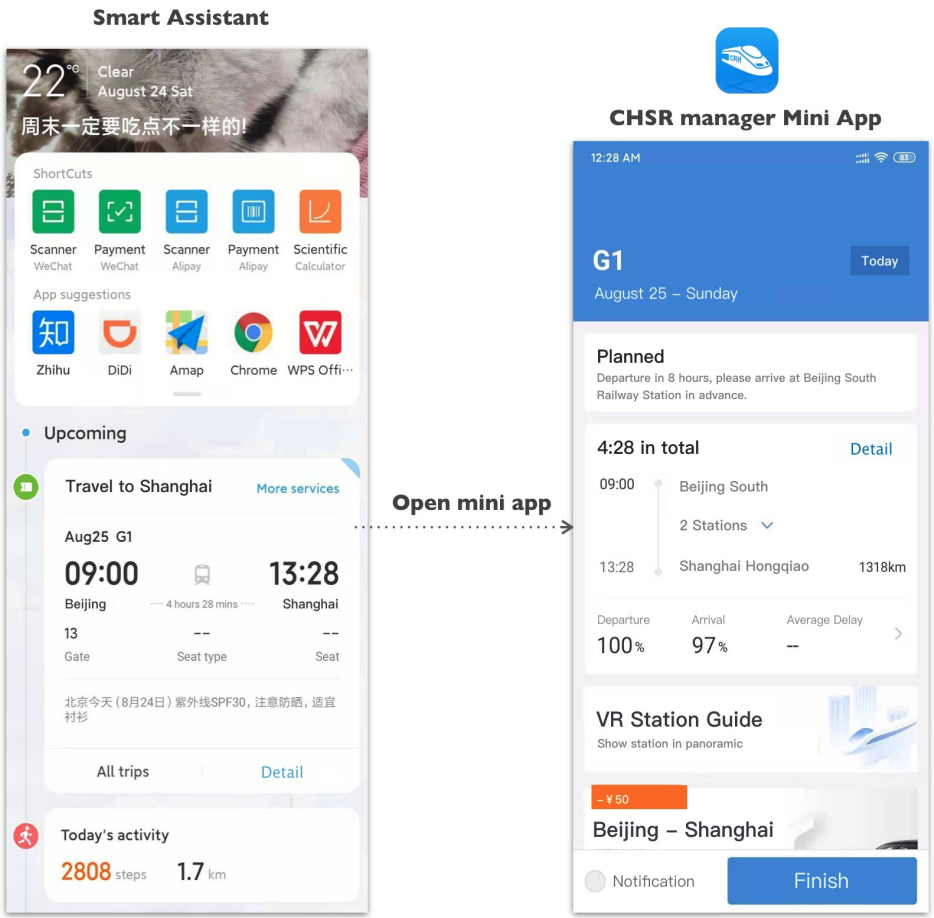
```
miniappuri    = uri-prefix uri-infix identify path-abempty ["?" query ] ["#" fragment ]

uri-prefix    = (custom-scheme "://") / ("https://" host)
custom-scheme = 1*unreserved
uri-infix     =  "miniapp"
identify      = id [";version=" version]
id            = 1*unreserved
version       = *unreserved
```

MiniApp Widget Requirements

Widget Requirement: https://w3c.github.io/miniapp-widget/req/

- MiniApp Widget 是 MiniApp 页面的一种特殊形式
- 与页面不同，Widget 可以仅占用屏幕的一个区域（而不是全部），用于为用户显示关键信息和响应简单的用户操作



MiniApp Test

Test: https://github.com/w3c/miniapp-tests

§ 2.1 Packaging

Id	Req	Title	Description	Specs	Ref
pkg-css-global-support	must	Global CSS stylesheet affects all pages	Global CSS stylesheet must affect all pages. The test includes CSS properties in the app.css over that will affect the text on the entry page, showing a black text on yellow background.	(1)	❐
pkg-pages-same-filenames	must	A page with resources with same filename	The entry page includes three resources (.html, .css, and .js) with the same filename. The app must be loaded.	(1)	❐
pkg-root-app-css-empty	may	Empty global app.css	The app.css global stylesheet may be empty. The app must be loaded.	(1)	❐

§ 2.3 Lifecycle

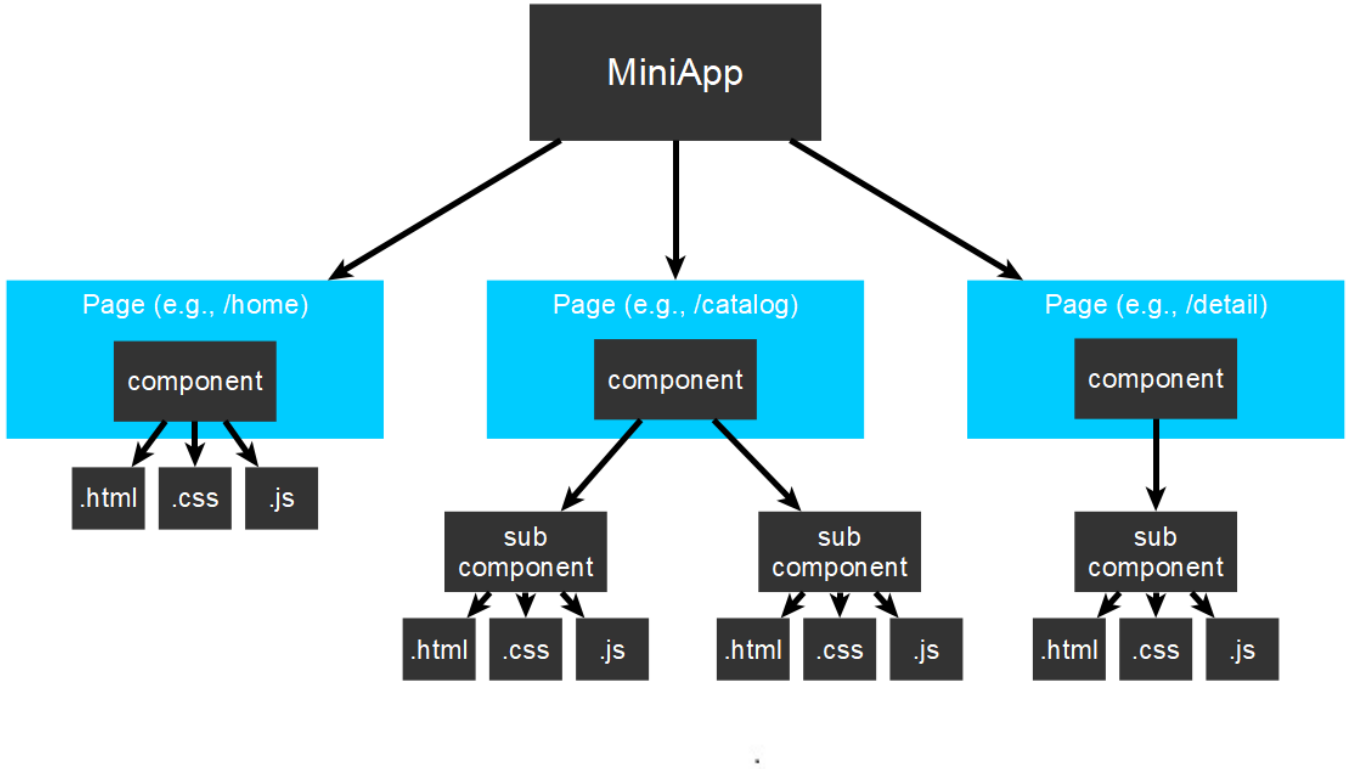
Id	Req	Title	Description	Specs	Ref
lcy-global-launched-callback-page-path	must	Global shown callback on the first page	Once the app is launched and the first page is shown, the callback must return an object. The pagePath member must be the same as the first page.	(1)	❐

§ 2.2 Manifest

Id	Req	Title	Description	Specs	Ref
mnf-window-background-color	must	Window background color set	The window's background member is set to #00FF00, so the app page must have a green background.	(1)	❐
mnf-window-background-color-default	must	Window background color set by default	If the window's background member is not set, the app page must have a white background by default.	(1)	❐
mnf-window-fullscreen-default	must	No full-screen by default	No window's full-screen member declared in the manifest, so ehe app must not be in full-screen mode.	(1)	❐
mnf-window-fullscreen-true	must	Fullscreen enabled in manifest	The window's fullscreen member is set to true in the manifest. The app must be shown in fullscreen.	(1)	❐
mnf-window-orientation-default	must	Portrait orientation by default	No window's orientation member declared in the manifest, so the app display must be in portrait mode by default.	(1)	❐
mnf-window-orientation-landscape	must	Landscape orientation defined in manifest	The window's orientation member is set to landscape in the manifest, so the app display must be in landscape mode.	(1)	❐
mnf-window-orientation-portrait	must	Portrait orientation defined in manifest	The window's orientation member is set to portrait in the manifest. The app display must be in portrait mode.	(1)	❐

MiniApp UI Components

Common UI Components: https://w3c.github.io/miniapp-components/



Basic Components

Basic components provide the minimum interactive blocks that can be reused for the MiniApp. It may include the following components such as 'Image', 'Progress', 'Text', 'Input', 'Button', 'Label', 'Select', 'Slider', 'Switch', 'Picker', 'Video', and 'Canvas' .

Container Components

Container components provide the structure of a MiniApp page. It may inculdes the following components such as 'Div', 'List', 'Swiper', 'Tabs', and 'Refresh'.

W3C MiniApp Standardization process

- Incubation
- Rec-track
- Publishing

W3C MiniApps CG: https://www.w3.org/community/miniapps/

Thanks