

Architektura frontendu aplikacji.gov.pl v 0.1

Streszczenie	3
Cel	3
Sposób realizacji	3
Konkluzja	3
Założenia	3
Implikacje założeń	4
Możliwe rozwiązania	4

Streszczenie

Cel

Opracowanie architektury frontendu platformy aplikacje.gov.pl.

Sposób realizacji

- Zebranie listy rozwiązań, które są zgodne z założeniami.
- Na podstawie analizy wad, wybór jednego rozwiązania ze stworzonej listy.

Konkluzja

- Poszczególne aplikacje umieszczane są w iframe'ach
- biblioteka app.js dołączana przez aplikacje wewnątrz ramki, komunikacja z aplikacją integrującą przez postMessage
- każda aplikacja rejestruje w aplikacji integrującej następujące informacje:
 - URL front-endu,
 - elementy menu,
- każda aplikacja rejestruje aplikacji Osobisty Pulpit widgety, jakie może udostępnić na Osobistym Pulpicie,
- osobisty pulpit jest również aplikacją, na tych samych prawach, co pozostałe.

Założenia

- Wszystkie zainstalowane na platformie aplikacje muszą mieć frontend zebrany w jednym miejscu. Istnieje jedna strona, z której można dostać się na strony poszczególnych aplikacji.
- Aplikacje mogą używać różnych frameworków JavaScript (szczególny przypadek: nie używać żadnych), a nawet różnych języków programowania (np. ClojureScript).
- Działający kod JS we frontendzie musi być izolowany pomiędzy aplikacjami. Jest to potrzebne, aby aplikacje nie mogły w niekontrolowany sposób zmieniać stanu innych aplikacji (przez przypadek czy też naumyślnie).
- Aplikacje powinny mieć możliwość komunikowania się między sobą (komunikacja asynchroniczna).
- Aplikacje nie powinny implementować wspólnych funkcji, takich jak menu Platformy, osobisty pulpit itp.

Implikacje założeń

- Zebranie frontendu w jednym miejscu: musi istnieć serwis, który konfiguruje się podając listę aplikacji, który będzie wyświetlał stronę listującą aplikacje. Są tutaj dwie główne możliwości:

- Aplikacje same serwują frontend: zamiary na nie to adresy URL ich interfejsów. (Najprostszy wariant „serwisu podającego listę aplikacji” to serwer HTTP serwujący stronę, na której jest lista linków.)
 - Aplikacje nie serwują frontentu: serwis podający listę aplikacji konfiguruje się podając mu opis tego, co każda aplikacja ma wyświetlać (HTML+JS+CSS). Tak skonfigurowany serwis serwuje frontend wszystkich aplikacji.
- Izolowanie wykonywanego kodu JS w przeglądarce można zrealizować za pomocą
 - uruchomienia kodu w innym oknie (np w iframe)
 - [WebWorkers](#)
 - zastosowanie rozwiązania, w którym aplikacje same serwują frontend na oddzielnych domenach
- Użycie różnych frameworków w różnych aplikacjach: zagadnienie jest pokrewne z izolowanie wykonywanego kodu. W tym przypadku izolować trzeba nie tylko zmienne ale także stan drzewa DOM. Poszczególne aplikacje powinny mieć możliwość modyfikacji tylko określonych fragmentów drzewa. Da się to zrealizować używając ramek (iframe). Kod działający w obrębie danej ramki może zmieniać tylko drzewo DOM tej ramki. Nie może zmieniać drzewa DOM ramek zagnieżdżonych. Możliwe, że da się takie ograniczenia zrealizować także w inny sposób, ale prawdopodobnie byłby on bardziej skomplikowany.
- Komunikacja między aplikacjami: w połączeniu z wymaganiami izolacji pozostaje jedynie możliwość komunikacji asynchronicznej.
 - W przypadku użycia ramek iframe można posłużyć się mechanizmem [postMessage](#).
- Aplikacje nie powinny implementować tej samej rzeczy wiele razy: aby uniknąć takiego wielokrotnego implementowania należy stworzyć bibliotekę (albo zestaw bibliotek), najlepiej niezależną od frameworku, która będzie dostarczać często używane funkcje.

Możliwe rozwiązania

Możliwość stosowania wielu frameworków i konieczność ograniczenia wpływu na drzewo DOM poprzez poszczególne aplikacje zmusza do wykorzystania iframe.

Pociąga to za sobą komunikację pomiędzy aplikacjami za pomocą postMessage.

Wydzielenie wspólnych części kodu do bibliotek jest dobrym i możliwym do zastosowania pomysłem niezależnie od przyjętego rozwiązania.

Do podjęcia pozostaje decyzja, w jaki sposób będzie działał serwis zbierający wszystkie frontнды.