PWA SUMMIT 2021

Bringing your PWA to App Stores

Micah Engle-Eshleman
Fullstack @ BeFunky

View talk here: youtu.be/qbh_u2hvljg?t=15600

Why App Stores?

- **Discovery**: users search for apps in stores

Why App Stores?

- **Discovery**: users search for apps in stores
- Higher **trust & familiarity** with app installation process

Why App Stores?

- **Discovery**: users search for apps in stores
- Higher **trust & familiarity** with app installation process
- Packaging your PWA is **relatively painless**. Reuse the same codebase but increase distribution.

Which App Stores?

- Play Store (Android/ChromeOS)
- Microsoft Store (Windows)
- Samsung Galaxy Store (Android)
- App Store* (iOS/Mac)

* requires significant refactoring

Google Play

Microsoft Store

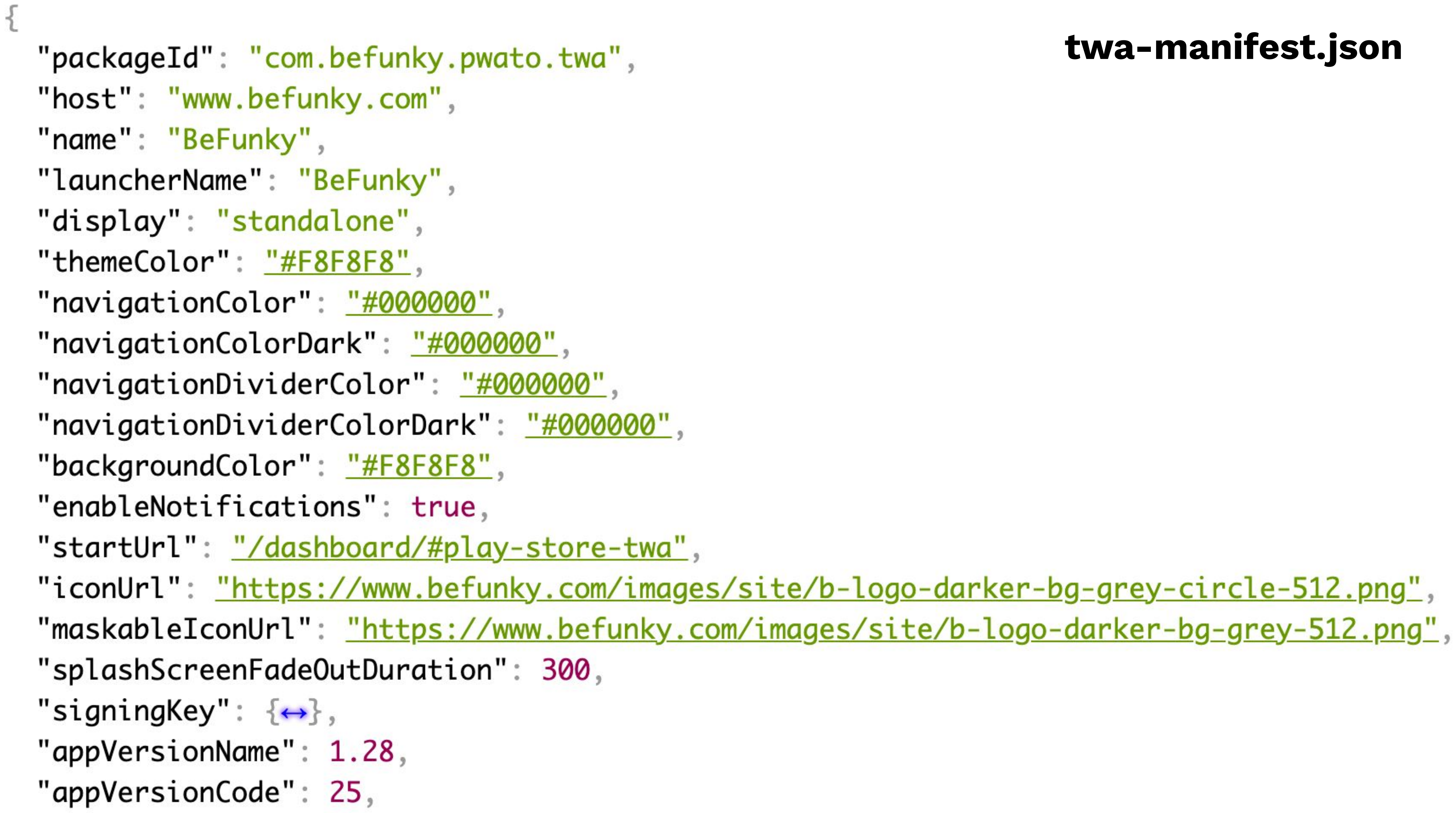
Galaxy Store

Quickstart

Try out pwabuilder.com

Packages your PWA for all supported app stores!

twa-manifest.json



```
{
  "packageId": "com.befunky.pwato.twa",
  "host": "www.befunky.com",
  "name": "BeFunky",
  "launcherName": "BeFunky",
  "display": "standalone",
  "themeColor": "#F8F8F8",
  "navigationColor": "#000000",
  "navigationColorDark": "#000000",
  "navigationDividerColor": "#000000",
  "navigationDividerColorDark": "#000000",
  "backgroundColor": "#F8F8F8",
  "enableNotifications": true,
  "startUrl": "/dashboard/#play-store-twa",
  "iconUrl": "https://www.befunky.com/images/site/b-logo-darker-bg-grey-circle-512.png",
  "maskableIconUrl": "https://www.befunky.com/images/site/b-logo-darker-bg-grey-512.png",
  "splashScreenFadeOutDuration": 300,
  "signingKey": {↔},
  "appVersionName": 1.28,
  "appVersionCode": 25,
```

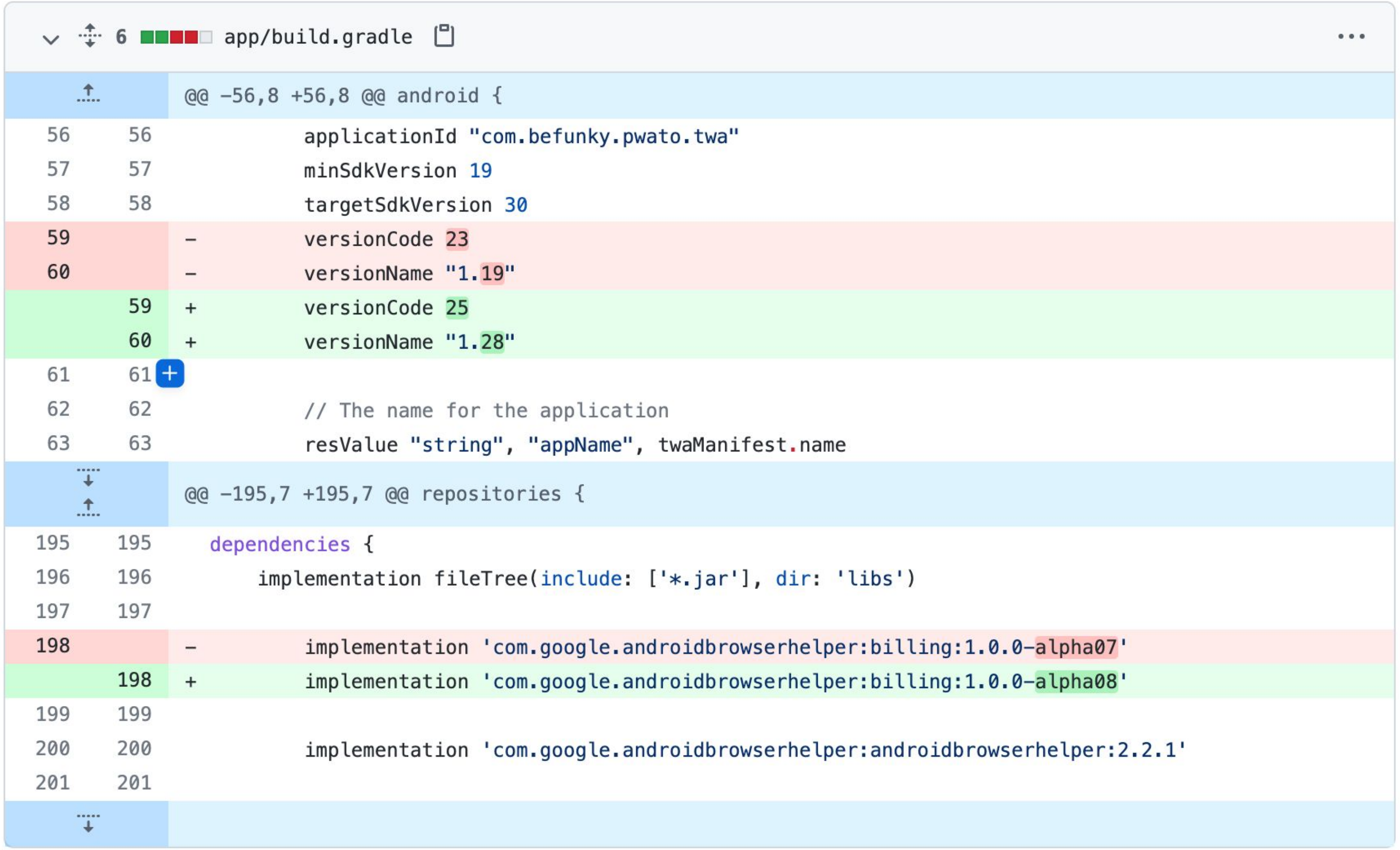
Bubblewrap CLI

Packages PWA for Play Store (as Trusted Web Activity)

```
$ npm install -g @bubblewrap/cli

$ bubblewrap init --manifest="manifest url..."

$ bubblewrap build
```

Quickstart article | Docs on Github

Version Control



6 app/build.gradle

```diff
@@ -56,8 +56,8 @@ android {
             applicationId "com.befunky.pwato.twa"
             minSdkVersion 19
             targetSdkVersion 30
-            versionCode 23
-            versionName "1.19"
+            versionCode 25
+            versionName "1.28"

             // The name for the application
             resValue "string", "appName", twaManifest.name
@@ -195,7 +195,7 @@ repositories {
 dependencies {
     implementation fileTree(include: ['*.jar'], dir: 'libs')

-        implementation 'com.google.androidbrowserhelper:billing:1.0.0-alpha07'
+        implementation 'com.google.androidbrowserhelper:billing:1.0.0-alpha08'

         implementation 'com.google.androidbrowserhelper:androidbrowserhelper:2.2.1'

```

Case Study

befunky.com

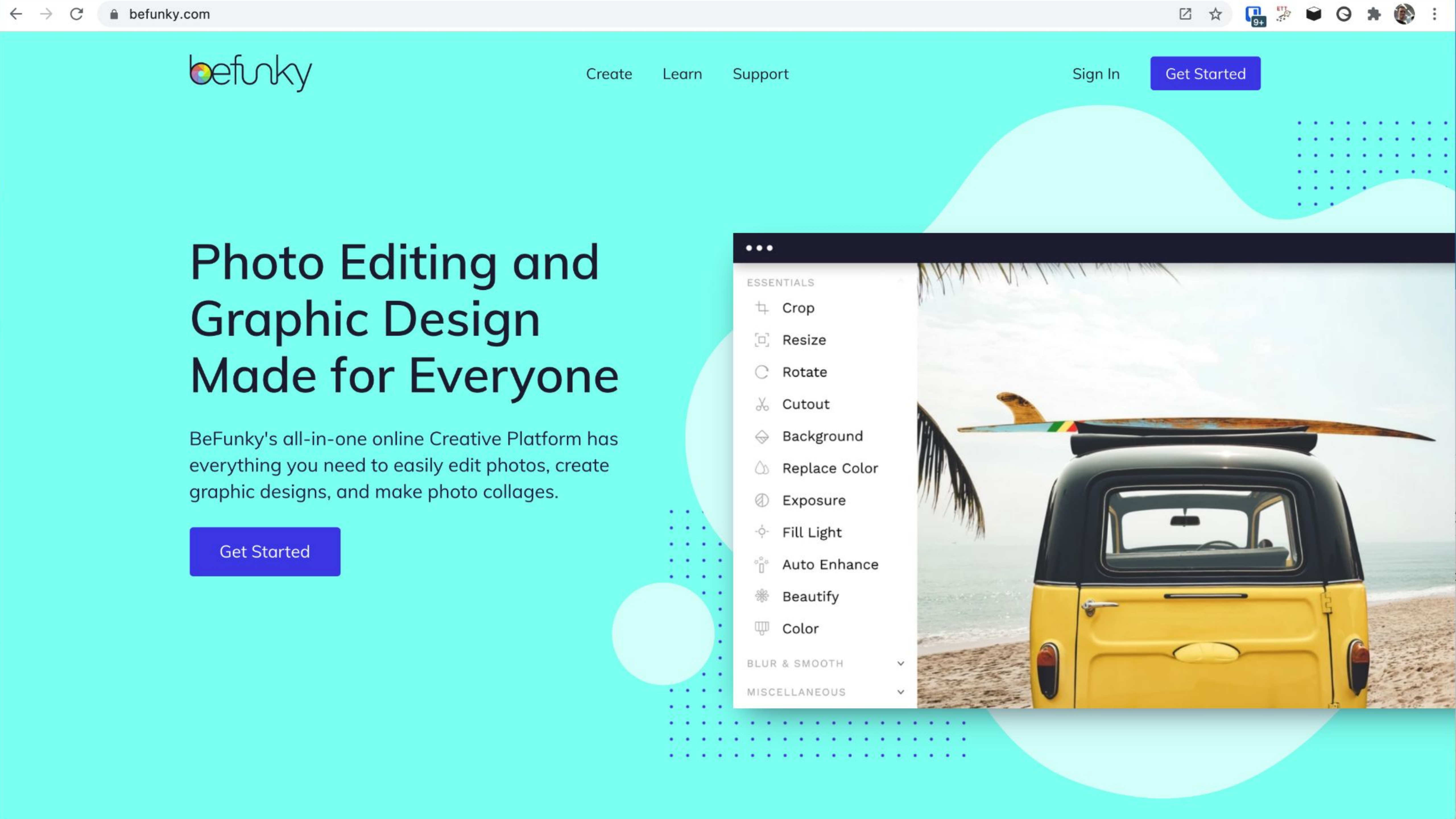


Photo Editing and Graphic Design Made for Everyone

BeFunky's all-in-one online Creative Platform has everything you need to easily edit photos, create graphic designs, and make photo collages.

Background

- **Frontend:** WebGL, Lit-html, Web Components

Background

- **Frontend:** WebGL, Lit-html, Web Components
- **Audience:** mostly desktop, some tablet (mobile coming soon)
  - 2-3 million monthly users
  - 6k monthly PWA users (**3x engagement**)

Background

- **Frontend:** WebGL, Lit-html, Web Components

- **Audience:** mostly desktop, some tablet (mobile coming soon)
  - 2-3 million monthly users
  - 6k monthly PWA users (**3x engagement**)

- **Why the Play Store?** ChromeOS is an increasingly popular desktop market

Which App Stores?

- Play Store (Android/ChromeOS)
- Microsoft Store (Windows)
- Samsung Galaxy Store (Android)
- App Store* (iOS/Mac)

* requires significant refactoring

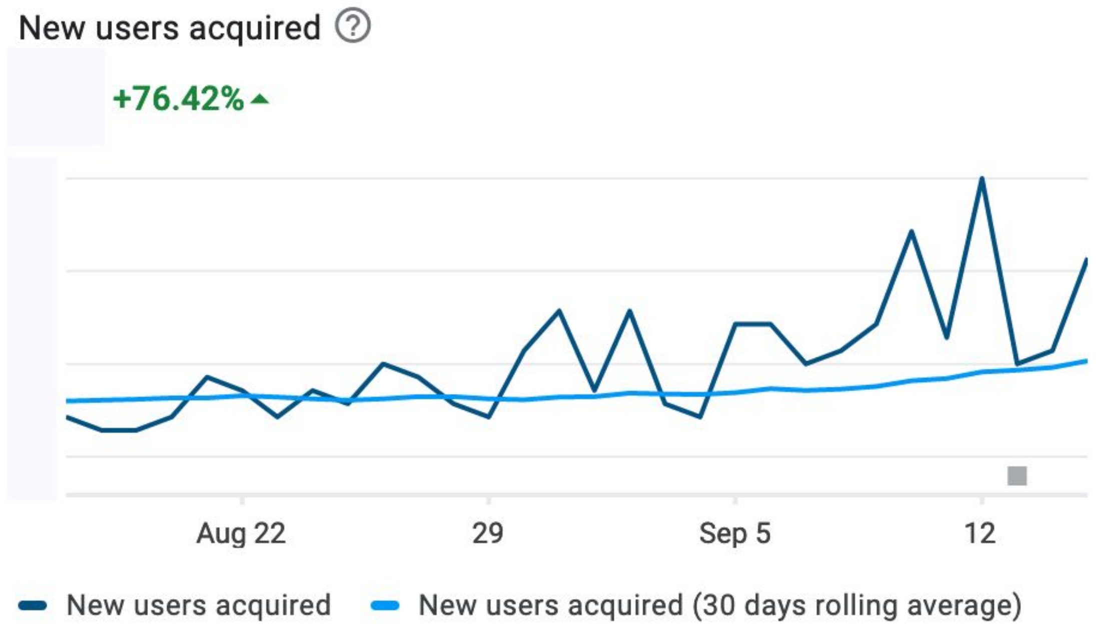
Google Play

Microsoft Store

Galaxy Store

Submitting PWA to Play Store

- Package PWA using Bubblewrap CLI
- Upload AAB file to Google Play Console
- Create new release (testing or production)



Analytics

How to know if PWA was installed from your website or via an app store?

- Customize start URL (in `twa-manifest.json`)
- Save flag in SessionStorage

Article: Bulletproof PWA & TWA Detection
micahjon.com/2021/pwa-twa-detection

Payments

How to charge users and keep track of subscriptions

App stores manage...

- Prices for products/subscriptions
- User’s access to products/subscriptions
- User’s payment methods
- Payment flow
- Renewal and cancellation

PWAs in the Play Store

- Prices for products/subscriptions
- User’s access to products/subscriptions

Digital Goods API

- User’s payment methods
- Payment flow

Payment Request API

- Renewal and cancellation

Digital Goods API

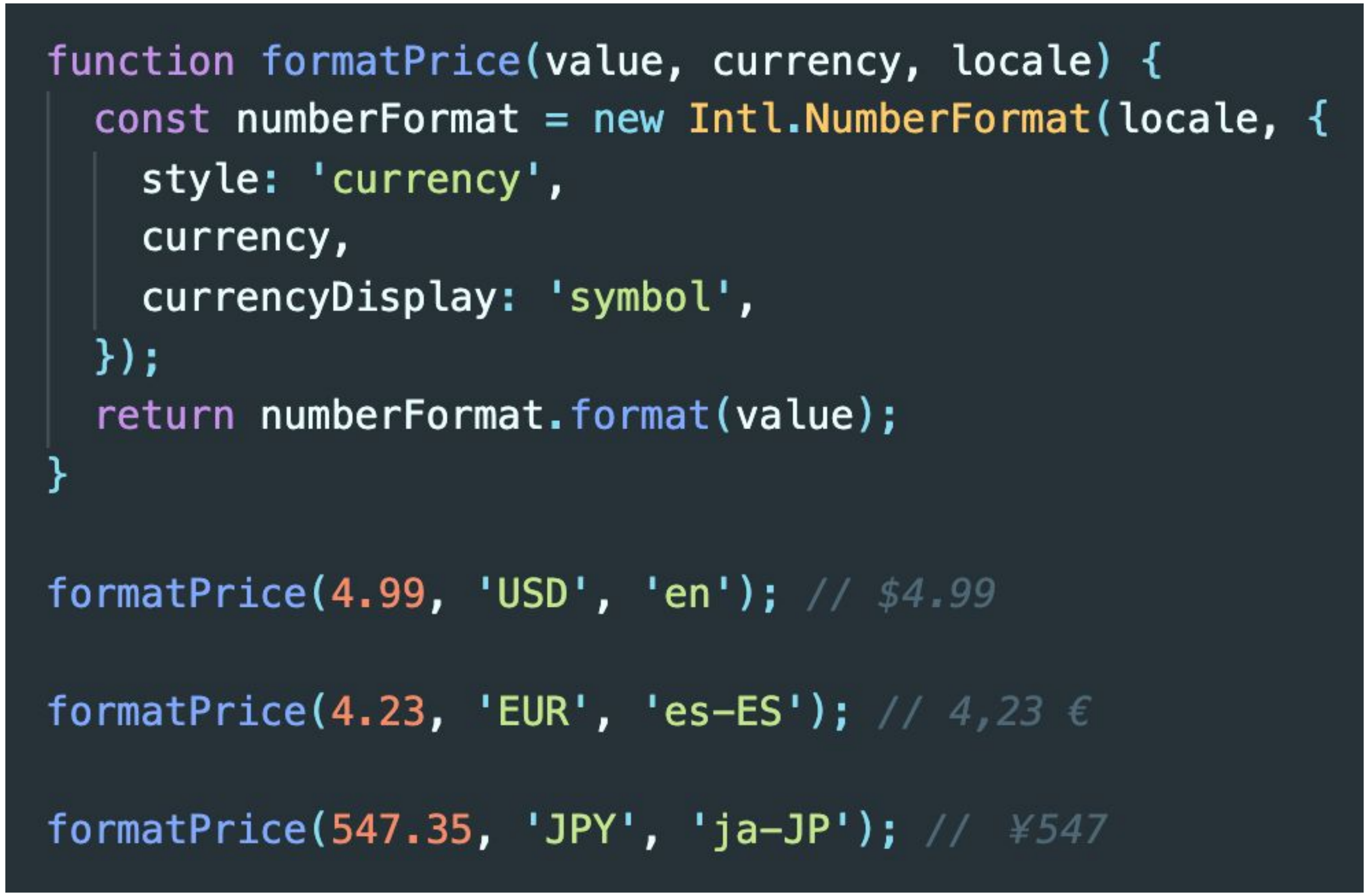
- Prices for products/subscriptions
- User’s access to products/subscriptions

Digital Goods API

```javascript
// Make Digital Goods API calls with Google Play Store service
const service = await window.getDigitalGoodsService("https://play.google.com/billing");

// Get the current user's purchases
const purchases = await service.listPurchases();

// Get subscription prices in user's preferred currency
const productSkus = ['plus_month_8_99_usd', 'plus_year_59_88_usd'];
const productDetails = await service.getDetails(productSkus);
```

Intl.NumberFormat

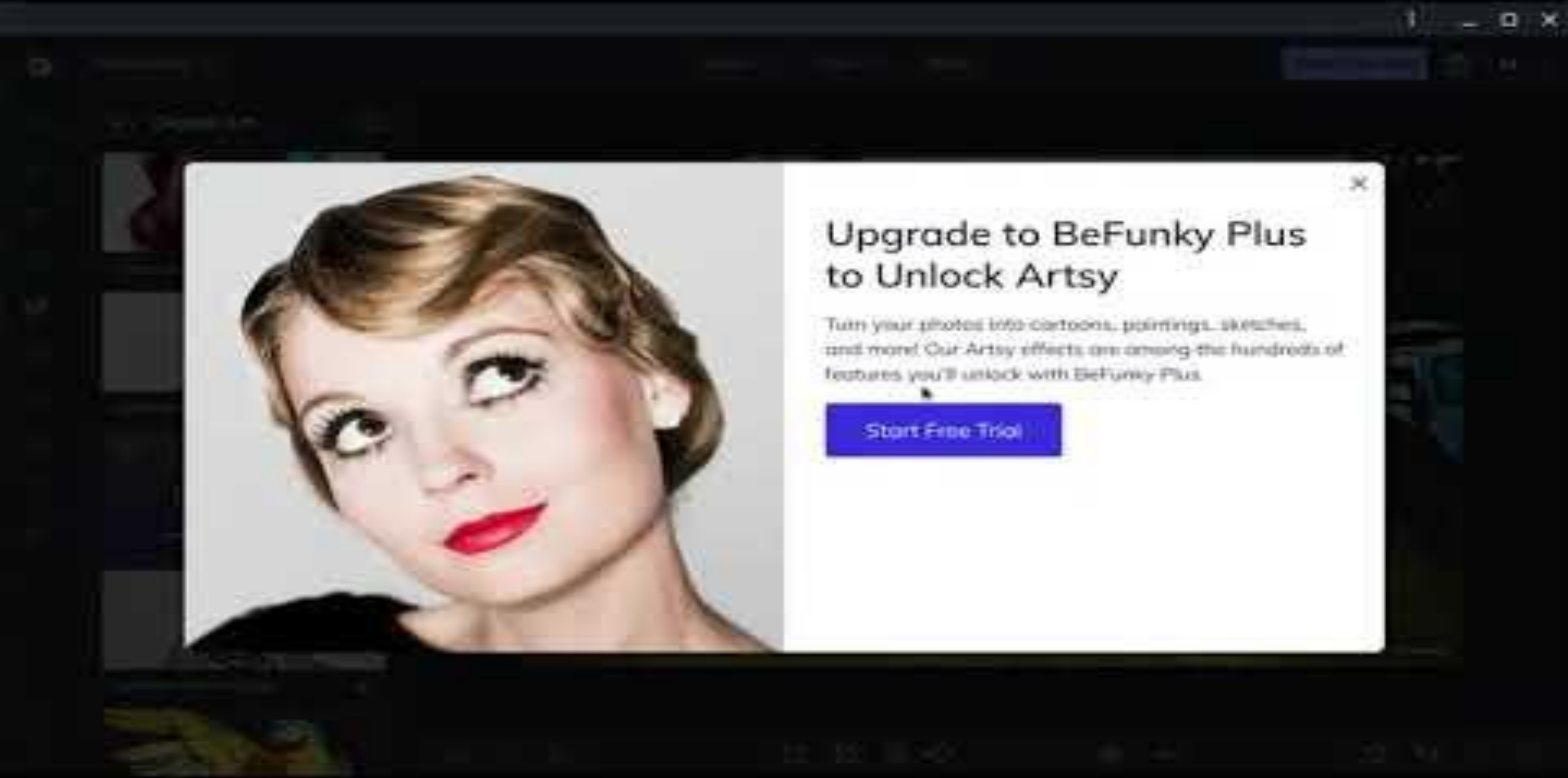
Display price in the user’s preferred currency and language



```
function formatPrice(value, currency, locale) {
  const numberFormat = new Intl.NumberFormat(locale, {
    style: 'currency',
    currency,
    currencyDisplay: 'symbol',
  });
  return numberFormat.format(value);
}

formatPrice(4.99, 'USD', 'en'); // $4.99

formatPrice(4.23, 'EUR', 'es-ES'); // 4,23 €

formatPrice(547.35, 'JPY', 'ja-JP'); //  ¥547
```

Payment Request API

- Prompts payment (in user’s preferred currency & locale)
- Processes payment (with user’s payment method)



Upgrade to BeFunky Plus to Unlock Artsy

Turn your photos into cartoons, paintings, sketches, and more! Our Artsy effects are among the hundreds of features you'll unlock with BeFunky Plus.

```
async function processPayment(sku) {

  const paymentRequest = new PaymentRequest([
    { supportedMethods: 'https://play.google.com/billing', data: { sku } },
  ]);

  // Open native dialog to prompt user to pay
  const paymentResponse = await paymentRequest.show();
  const { purchaseToken } = paymentResponse.details;

  // Validate payment on backend (using Google Play API) and upgrade user
  await BeFunky.withLoadScreen(BeFunky.validatePlayStorePayment(sku, purchaseToken));

  // Acknowledge purchase with Digital Goods API. Otherwise, user will get refund in 3 days.
  const service = await window.getDigitalGoodsService('https://play.google.com/billing');
  await service.acknowledge(purchaseToken, 'onetime');

  // Payment was successful!
  await paymentResponse.complete('success');

}
```

```
async function processPayment(sku) {

  const paymentRequest = new PaymentRequest([
    { supportedMethods: 'https://play.google.com/billing', data: { sku } },
  ]);

  // Open native dialog to prompt user to pay
  const paymentResponse = await paymentRequest.show();
  const { purchaseToken } = paymentResponse.details;

  // Validate payment on backend (using Google Play API) and upgrade user
  await BeFunky.withLoadScreen(BeFunky.validatePlayStorePayment(sku, purchaseToken));

  // Acknowledge purchase with Digital Goods API. Otherwise, user will get refund in 3 days.
  const service = await window.getDigitalGoodsService('https://play.google.com/billing');
  await service.acknowledge(purchaseToken, 'onetime');

  // Payment was successful!
  await paymentResponse.complete('success');

}
```

Show Prompt

```
async function processPayment(sku) {

  const paymentRequest = new PaymentRequest([
    { supportedMethods: 'https://play.google.com/billing', data: { sku } },
  ]);

  // Open native dialog to prompt user to pay
  const paymentResponse = await paymentRequest.show();
  const { purchaseToken } = paymentResponse.details;

  // Validate payment on backend (using Google Play API) and upgrade user
  await BeFunky.withLoadScreen(BeFunky.validatePlayStorePayment(sku, purchaseToken));

  // Acknowledge purchase with Digital Goods API. Otherwise, user will get refund in 3 days.
  const service = await window.getDigitalGoodsService('https://play.google.com/billing');
  await service.acknowledge(purchaseToken, 'onetime');

  // Payment was successful!
  await paymentResponse.complete('success');

}
```

Validate Payment

```javascript
async function processPayment(sku) {

  const paymentRequest = new PaymentRequest([
    { supportedMethods: 'https://play.google.com/billing', data: { sku } },
  ]);

  // Open native dialog to prompt user to pay
  const paymentResponse = await paymentRequest.show();
  const { purchaseToken } = paymentResponse.details;

  // Validate payment on backend (using Google Play API) and upgrade user
  await BeFunky.withLoadScreen(BeFunky.validatePlayStorePayment(sku, purchaseToken));

  // Acknowledge purchase with Digital Goods API. Otherwise, user will get refund in 3 days.
  const service = await window.getDigitalGoodsService('https://play.google.com/billing');
  await service.acknowledge(purchaseToken, 'onetime');

  // Payment was successful!
  await paymentResponse.complete('success');

}
```

Acknowledge Purchase

```javascript
async function processPayment(sku) {

  const paymentRequest = new PaymentRequest([
    { supportedMethods: 'https://play.google.com/billing', data: { sku } },
  ]);

  // Open native dialog to prompt user to pay
  const paymentResponse = await paymentRequest.show();
  const { purchaseToken } = paymentResponse.details;

  // Validate payment on backend (using Google Play API) and upgrade user
  await BeFunky.withLoadScreen(BeFunky.validatePlayStorePayment(sku, purchaseToken));

  // Acknowledge purchase with Digital Goods API. Otherwise, user will get refund in 3 days.
  const service = await window.getDigitalGoodsService('https://play.google.com/billing');
  await service.acknowledge(purchaseToken, 'onetime');

  // Payment was successful!
  await paymentResponse.complete('success');

}
```

Tell browser that payment succeeded (closes native payment prompt)

Care about PWAs and web performance?

Come work with us!

Thanks!

You can find me at on the web at [micahjon.com]

[@micahjon] on Github, [@micahjme] on Twitter

Try out BeFunky at [befunky.com/create]

Thanks to Alexander Nohe and Sam Richard at Google for their encouragement and advice!