Changes in the Payment Request API

By Eiji Kitamura
Developer Advocate in Tokyo

Since the launch of the Payment Request API in Chrome 53, a few changes have been made to the API. These changes won't break the functionalities of your working code, but we recommend you to add a shim to your code so that future changes won't break your product.

Note: All changes described here are already reflected in the existing integration guides.

Chrome 62

PaymentDetailsModifier is now available

In a payment request, there are cases where you want to provide a discount or an extra charge depending on the payment method a customer chooses. `PaymentDetailsModifier` is the feature you can use to achieve this.

Add `modifiers` property to the second argument of `PaymentRequest` constructor along with an array of `PaymentDetailsModifier` object which is declarative rules of how display items and total should be modified depending on the payment method of the customer's choice.

Following example shows how you declare a user to be charged $3 processing fee for choosing a credit card payment.

```
let methods = [{
 supportedMethods: 'basic-card',
 data: {
   supportedNetworks: ['visa', 'mastercard']
   supportedTypes: ['credit', 'debit']
 }
}];

let details = {
 displayItems: [{
   label: 'T-shirt',
   amount: { currency: 'USD', value: '68.00' }
 }],
 total: {
   label: 'Total price',
   amount: { currency: 'USD', value: '68.00' }
 },
 modifiers: [{
   supportedMethods: 'basic-card',
   data: {
     supportedTypes: ['credit']
   },
   additionalDisplayItems: [{
     label: 'Processing fee',
     amount: { currency: 'USD', value: '3.00' }
   }],
   total: {
     label: 'Credit card price',
     amount: {currency: ‘USD’, value: ‘71.00’}}
 }]
};

let options = {};

let request = new PaymentRequest(methods, details, options);
```

On actual Payment Request sheet, as soon as picking a credit card payment, a user will see additional display item called "Processing fee" with $3 charge, with total price of $71.00.

`PaymentDetailsModifier` contains following parameters:

Property name	
`supportedMethods`	Specify which payment method applies this rule.
`additionalDisplayItems`	Additional display items you'd like to add either discounts or extra charges.
`total`	Total price after adding the additionalDisplayItems.
`data`	For **`basic-card`** payment, this will be used as a way to filter specific card types with **`supportedTypes`**. Otherwise the usage of this parameter depends on the payment method (payment app). Refer to the documentation each payment method provides.

When there's shipping options, things get a bit tricky, because `modifiers`' total price can not be static and needs dynamic modification.

To achieve this, you will modify `modifiers` property's `additionalDisplayItems` and `total` upon `shippingaddresschange` and `shippingoptionchange` event just like you do to `displayItems` property of `PaymentDetails` object.

supportedMethods property now takes a string

`supportedMethods` property in `PaymentMethodData` object (the first argument to the `PaymentRequest` constructor) has been taking an array of strings that indicates a payment method. It's now taking a single string as an argument.

Note that giving an array will continue to work for the time being.

✓ **Old** - still works for the time being.

```
var methodData = [{
  supportedMethods: ['basic-card'],
  data: {
    supportedNetworks: ['visa', 'mastercard', 'amex', 'jcb',
      'diners', 'discover', 'mir', 'unionpay']
  }
}, {
  supportedMethods: ['https://bobpay.xyz']
}];
```

✓ **New** - the new way.

```
var methodData = [{
  supportedMethods: 'basic-card',
  data: {
    supportedNetworks: ['visa', 'mastercard', 'amex', 'jcb',
      'diners', 'discover', 'mir', 'unionpay']
   }
}, {
  supportedMethods: 'https://bobpay.xyz'
}];
```

Chrome 61

supportedTypes in basic card is available

When `supportedMethods` is '`basic-card`', you can now provide `supportedTypes` to indicate which type of cards are supported among '`credit`', '`debit`', and '`prepaid`'.

```
var methodData = [{
 supportedMethods: 'basic-card',
 data: {
   supportedNetworks: ['visa', 'mastercard', 'amex', 'jcb',
     'diners', 'discover', 'mir', 'unionpay']
   supportedTypes: ['credit', 'debit', 'prepaid']
 }
}];
```

Make sure to double check the card type with your payment gateway as this filtering may not work perfectly depending on where it's sourced from.

Chrome 57

PaymentRequest is now available inside iframes

The Payment Request API can now be called from within an `iframe` by adding the `allowpaymentrequest` attribute to the `iframe` element.

```
<iframe src="/totally/fake/url" allowpaymentrequest></iframe>
```

PaymentMethodData supports "basic-card"

The first argument to `PaymentRequest()` constructor is an array of payment method data. The `PaymentMethodData` format has been altered in this release.

✓ **Old** - still works for the time being.

```
var methodData = [{
  supportedMethods: ['visa', 'mastercard', 'amex', 'jcb']
}];
var request = new PaymentRequest(methodData, details, options);
```

✓ **New** - the new structure.

```
var methodData = [{
  supportedMethods: ['basic-card'],
  data: {
    supportedNetworks: ['visa', 'mastercard', 'amex', 'jcb',
      'diners', 'discover', 'mir', 'unionpay']
  }
}];
var request = new PaymentRequest(methodData, details, options);
```

The format of the **data** property depends on the value in **supportedMethods** and is based on the Basic Card specification. Note that the spec includes **supportedTypes** which accepts **credit**, **debit** or **prepaid**, but Chrome 57 ignores this property and treats any values in **supoprtedNetworks** as credit cards.

```
var methodData = [{
  supportedMethods: ['basic-card'],
  data: {
    supportedNetworks: ['visa', 'mastercard', 'amex', 'jcb',
      'diners', 'discover', 'mir', 'unionpay'],
    supportedTypes: ['credit'] <= not available
  }
}];
```

Chrome 56

pending

As part of payment item information, developers can add **pending** to indicate that the price is not fully determined yet. The **pending** field accepts a boolean value.

```
{
  label: "State tax",
  amount: { currency: "USD", value : "5.00" },
  pending: true
},
```

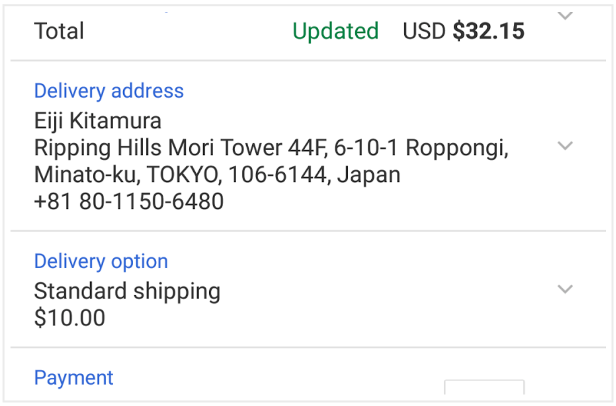
This is commonly used to show line items such as shipping or tax amounts that depend on selection of shipping address or shipping options. Chrome indicates pending fields in the UI for the payment request.

requestPayerName

As part of shipping option (third argument to **PaymentRequest**), developers can now add **requestPayerName** to request the payer's name separate from shipping address information. **requestPayerName** accepts a boolean value.

shippingType

As part of shipping option (third argument to **`PaymentRequest`**), developers can now add **`shippingType`** to request that the UI show "delivery" or "pickup" instead of "shipping". **`shippingType`** accepts the strings **`shipping`** (default), **`delivery`**, or **`pickup`**.

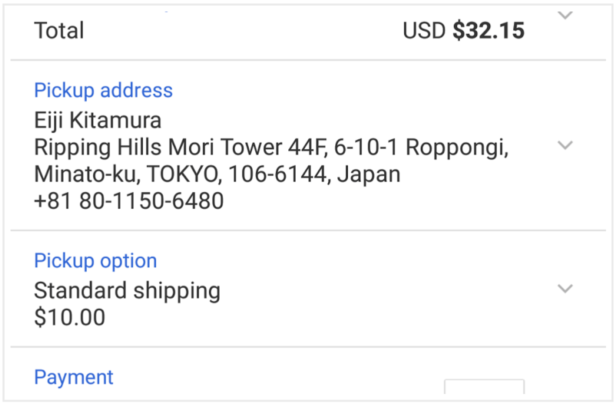

Total Updated USD **$32.15**

Delivery address
Eiji Kitamura
Ripping Hills Mori Tower 44F, 6-10-1 Roppongi,
Minato-ku, TOKYO, 106-6144, Japan
+81 80-1150-6480

Delivery option
Standard shipping
$10.00

Payment

shippingType="delivery"


Total USD **$32.15**

Pickup address
Eiji Kitamura
Ripping Hills Mori Tower 44F, 6-10-1 Roppongi,
Minato-ku, TOKYO, 106-6144, Japan
+81 80-1150-6480

Pickup option
Standard shipping
$10.00

Payment

shippingType="pickup"

Serializer functions available to PaymentResponse and PaymentAddress

PaymentResponse object and **`PaymentAddress`** object are now JSON-serializable. Developers can convert one to a JSON object by calling the **`toJSON()`** function and avoid creating cumbersome **`toDict()`** functions.

```
request.show().then(response => {
  let res = response.toJSON();
});
```

canMakePayment

In addition to the API availability, you can check to see if a user has an active payment method before invoking the Payment Request API. Remember that this is optional as users can still add a new payment method on the payment UI.

```
let request = new PaymentRequest(methods, details, options);
if (request.canMakePayment) {
  request.canMakePayment().then(result => {
    if (result) {
      // Payment methods are available.
    } else {
      // Payment methods are not available, but users can still add
      // a new payment method in Payment UI.
    }
  }).catch(error => {
    // Unable to determine.
  });
}
```

Last updated July 2, 2018.