React Web Event Code Samples

The following events are available to use.

Metadata Event

Imports `LanguageData` from `@personalization-js-sdk/common`.

```
interface LanguageData {  language: string;}
```

Example

```
import React, {useEffect} from 'react';import Personalization from "@personalization-js-sdk/web"import { View , Text} from "web" ; import { LangaugeData ,EventTypes } from '@personalization-js-sdk/common';const LangaugeDataInitial : LangaugeData = {  language:"en-GB"};const App = () => {    const PersonalizationInstance = new Personalization(account, user);      useEffect = () => {  PersonalizationInstance.report (    EventTypes.ContextMetadata, {metadata: LangaugeDataInitial}  ),  },[];      return <View>    <Text> MonetateSDK Metadata Event.</Text>  </View>}export default App;
```

Screen Size Event

Imports `ScreenSizeData` from `@personalization-js-sdk/common`.

```
interface ScreenSizeData {  height: number;  width: number;}
```

Example

```
import React, {useEffect} from 'react';import { View , Text} from "web";import { ScreenSizeData, EventTypes} from '@personalization-js-sdk/common';import Personalization from "@personalization-js-sdk/web";const ScreenSizeDataInitial : ScreenSizeData = {  height: 250;  width: 500;};const App = () => {     const PersonalizationInstance = new Personalization(account, user);      useEffect(() => {  PersonalizationInstance.report (    EventTypes.ContextScreenSize, screenSize: {...ScreenSizeDataInitial}  ),  },[];      return <View>    <Text> MonetateSDK Screen Size Event.</Text>  </View>   }export default App;
```

User Agent Event

Imports `UserAgentData` from `@personalization-js-sdk/common`.

```
interface UserAgentData {   userAgent: string;}
```

Example

```
import React, {useEffect} from 'react';import { View , Text} from "web";import DeviceInfo
from 'web-device-info';import Personalization from "@personalization-js-sdk/web"import {
UserAgentData, EventTypes } from '@personalization-js-sdk/common';const
UserAgentDataDataInitial : UserAgentData = {   userAgent: "userAgent as string"};const App
= () => {     const PersonalizationInstance = new Personalization(account, user);
useEffect(() => {     const deviceName = await DeviceInfo.getDeviceName();     const
userAgent =
`${DeviceInfo.getApplicationName()}/${DeviceInfo.getVersion()}/${DeviceInfo.getSystemName
;   };     PersonalizationInstance.report (     EventTypes.ContextUserAgent, {userAgent:
userAgent}    ),   },[])     return <View>     <Text> MonetateSDK User Agent event.</Text>
</View> }export default App;
```

Coordinates Event

Reports the coordinates of the customer.

```
interface CoordinatesData {   latitude: string;   longitude: string;}
```

Example

```
import React, {useEffect} from 'react';import { View , Text} from "web";import
Personalization from "@personalization-js-sdk/web"import { CoordinatesData, EventTypes }
from '@personalization-js-sdk/common';const CoordinatesInitialData : CoordinatesData = {
latitude: "11.2734",  longitude: "45.2734"};const App = () => {      const
PersonalizationInstance = new Personalization(account, user);      useEffect(()=> {  // for
manual use     PersonalizationInstance.report (     EventTypes.ContextCoordinates,
{coordinates: CoordinatesInitialData}    ),  },[]);      return <View>     <Text> MonetateSDK
Coordinate event.</Text>   </View>}export default App;
```

Referrer Event

Reports the referrer.

```
interface ReferrerData {   referrer: string;}
```

Example

```
import React, {useEffect} from 'react';import { View , Text} from "web";import Personalization from "@personalization-js-sdk/web"import { ReferrerData,EventTypes } from '@personalization-js-sdk/common';const UserAgentDataDataInitial : ReferrerData = { referrer: "refer"};const App = () => {    const PersonalizationInstance = new Personalization(account, user);       useEffect(()=> {  PersonalizationInstance.report (   EventTypes.ContextReferrer, { referrer: referrer}   )  }, []);       return <View>     <Text> MonetateSDK Referrer Event.</Text>  </View>}export default App;
```

Context Variable Event

Imports `ContextVariablesData` from `@personalization-js-sdk/common`, and reports custom variables.

```
interface ContextVariablesData {  variable: string;  value: string;}interface CustomVariablesData {  customVariables: Array<ContextVariablesData>;}
```

Example

```
import  {useEffect} from 'react';import { View , Text} from "web";import Personalization from "@personalization-js-sdk/web";import { CustomVariablesData, EventTypes } from '@personalization-js-sdk/common';const customVariablesInitialData : CustomVariablesData = {  customVariables = [{    variable: "x",    value: "0.1"  }] };const App = () => {    const PersonalizationInstance = new Personalization(account, user);       useEffect(() => {  PersonalizationInstance.report (    EventTypes.ContextCustomVariables, {customVariables: customVariablesInitialData}  ),  }.[]);       return <View>    <Text> MonetateSDK Custom Variables Event.</Text>  </View>}export default App;
```

Page Event

Reports any events on the page.

```
interface PageEventsData {  pageEvents: string[];}
```

Example

```
import React,{useEffect} from 'react';import { View , Text} from "web";import
Personalization from "@personalization-js-sdk/web"import { PageEventsData, EventTypes}
from '@personalization-js-sdk/common';const PageEventsDataInitial : PageEventsData = {
pageEvents:["MyEvent"]};const App = () => {     const PersonalizationInstance = new
Personalization(account, user);      useEffect(() => {  PersonalizationInstance.report (
EventTypes.RecordPageEvents, {pageEvents: PageEventsDataInitial}  );  }, []);       return
<View>    <Text> MonetateSDK Page Event.</Text>  </View>}export default App;
```

Page View Event

Reports when a customer views a specified page.

```
interface PageViewData {  url: string;  pageType: string;  categories: string[];  breadcrumbs:
string[];}
```

Example

```
import React, {useEffect} from 'react';import { View , Text} from "web" ;import
Personalization from "@personalization-js-sdk/web"import { PageViewData, EventTypes }
from '@personalization-js-sdk/common';const PageViewDataInitial : PageViewData = {  url:
"www.monetate.com";  pageType: "cart";  categories: ["category"];  breadcrumbs:
["breadcrumbs"];};const App = () => {     const PersonalizationInstance = new
Personalization(account, user);      useEffect(()=> {  PersonalizationInstance.report (
EventTypes.ContextPageView, {...PageViewDataInitial}  ),  },[]);      return <View>    <Text>
MonetateSDK Page View Event.</Text>  </View>}export default App;
```

Product Thumbnail Event

Reports when a customer views a product thumbnail.

```
interface ProductThumbnailViewData {  products: string[];}
```

Example

```
import React, {useEffect} from 'react';import { View , Text} from "web" ; import
Personalization from "@personalization-js-sdk/web"import {
ProductThumbnailViewData,EventTypes } from '@personalization-js-sdk/common';const
ProductThumbnailViewDataInitial : ProductThumbnailViewData = {  products: ["product-1"
,Product-2];};const App = () => {     const PersonalizationInstance = new
Personalization(account, user);      useEffect(() => {  PersonalizationInstance.report (
EventTypes.ContextProductThumbnailView, {...ProductThumbnailViewDataInitial}  ),  },[]);
return <View>     <Text>MonetateSDK Product Thumbnail View Event.</Text>
</View>}export default App;
```

Product Detail View Event

Reports when a customer views a product detail page.

```
interface Product {  productId: string;  sku: string;}interface ProductDetailViewData {
products: Product[];}
```

Example

```
import React, {useEffect} from 'react';import { View , Text} from "web" ; import
Personalization from "@personalization-js-sdk/web"import { ProductDetailViewData,
EventTypes } from '@personalization-js-sdk/common';const ProductDetailViewDataInitial :
ProductDetailViewData = {  products: [{    productId: "",    sku: ""  }] };const App = () => {
const PersonalizationInstance = new Personalization(account, user);       useEffect(() => {
PersonalizationInstance.report (    EventTypes.ContextProductDetailView,
{...ProductDetailViewDataInitial}  ),  },[]);       return <View>    <Text> MonetateSDK Page
View Event.</Text>  </View>}export default App;
```

Recommendation Clicks Event

Reports recommendations click events.

```
interface RecClicksData {  recClicks: Array<string>;}
```

Example

```
import React, {useState} from 'react';import { View , Text} from "web";import
Personalization from "@personalization-js-sdk/web";import { RecClicksData, EventTypes }
from '@personalization-js-sdk/common'; const RecClicksInitialData : RecClicksData = {
recClicks: [],};const RecClickScreen = () => {const PersonalizationInstance = new
Personalization(account, user);const [recState, updateRecState] =
useState(RecClicksInitialData);  const handleRecClicks = (recClick: string) => {
updateRecState(prevState => {      return {        ...prevState,        recClicks:
[...recState.recClicks, recClick]      }     });  };      const reportRecClicks = () => {
PersonalizationInstance.report(      EventTypes.RecordRecClicks, {recClicks :
recState.recClicks}    ),  }    return <View>    <View>      <TouchableNativeFeedback
onPress={() => handleRecClicks("sjscjcffuedscldm")}>        // Show List of Products.
<Image src={// Enter  image source }/>        <Text> {// Show title}</Text>
</TouchableNativeFeedback>    </View>    <Button title = "report Clicks" onPress =
{()=>reportRecClicks() }/>  </View>}export default RecClicksScreen;
```

Recommendation Impressions Event

Reports recommendations impression events.

```
interface RecClicksData {  recImpressions: Array<string>;}
```

Example

```
import React, {useState} from 'react';import { View, Text, Image, TouchableNativeFeedback,
Button } from 'web';import Personalization from "@personalization-js-sdk/web";import {
RecClicksData, EventTypes } from '@personalization-js-sdk/common';const
RecImpressionsDataInitial : RecImpressionsData = {  recImpressions: [],};const
RecImpressionsScreen = () => {const PersonalizationInstance = new
Personalization(account, user);const [recState, updateRecState] =
useState(RecImpressionsDataInitial);const handleRecrecImpressions = (recImpressions:
string) => {  updateRecState(prevState => {    return {      ...prevState,      recImpressions:
[...recState.recImpressions, recImpressions]    }  });};    const reportRecImpressions = () =>
{      PersonalizationInstance.report(    EventTypes.RecordRecImpressions, {recImpressions :
recState.recImpressions}  ),}  return <View>  <View>    <TouchableNativeFeedback
onPress={() => handleRecImpressions("sjscjcffffkdfnkduedscldm")}>      // Show List of
Products.      <Image src={// Enter  image source }>      <Text> {// SHow title}</Text>
</TouchableNativeFeedback>        </View>  <Button title = "report Impressions" onPress =
{()=>reportRecImpressions() }/></View>}export default RecImpressionsScreen;
```

Impressions Event

Reports impression events.

```
interface ImpressionsData {  impressionIds: Array<string>;}
```

Example

```
import React, {useEffect} from 'react';import { View , Text} from "web" ;import Personalization from "@personalization-js-sdk/web"import { ImpressionsData, EventTypes } from '@personalization-js-sdk/common';const ImpressionsDataInitial : ImpressionsData = {  impressionIds:["1","2","3"] // Array of strings Impression IDs is required};const App = () => {  const PersonalizationInstance = new Personalization(account, user);      useEffect(() => {    PersonalizationInstance.report (    EventTypes.RecordImpressions, {impressionIds: ImpressionsDataInitial}   ),  } ,[]);      return <View>    <Text> MonetateSDK Impression Event.</Text>  </View>}export default App;
```

Add to Cart Event

Reports when a customer adds a product to their cart. Import and use the `CartLinesData` interface for this event. `CartLine` is the data structure used by this interface.

```
interface CartLine {  sku: string;  pid: string;  quantity: number;  currency: string;  value: string;}interface CartLinesData {  cartLines: Array<CartLine>;}
```

Example

```
import React,{ useState, useEffect} from 'react';import Personalization from "@personalization-js-sdk/web" import { CartData, EventTypes } from '@personalization-js-sdk/common';import { View , Text} from "web" ; const CartLinesInitialData: CartData = {  cartLines: []};const App = () => {  const [CartState, updateCartState] = useState(CartLinesInitialData);  const PersonalizationInstance = new Personalization(account, user);    useEffect = () => {    AddToCartItemsDetails: {      sku: "T-shirt",      pid: "1",      quantity: 1,      currency: "usd",      value: "30"    }    updateCartState(prevState => {      return {        ...prevState,        cartLines: [...updateCartState.cartLines, AddToCartItemsDetails]      }    });  }  const reportAddToCart = () => {    PersonalizationInstance.report(      EventTypes.ContextCart, { cartLines: [...this.state.CartState] }    ),  }  return <View>    <View>      // Product Details      </View>    <Button title = "Add to cart " onPress ={()=>reportAddToCart() }/>    </View>}export default App;
```

Purchase Event

Reports when a customer purchases products. Import and use the `PurchaseData` interface for this event.

PurchaseLine is the data structure used by this interface.

```
interface PurchaseLine {  sku: string;  pid: string;  quantity: number;  currency: string;  value: string;}interface PurchaseData {  account?: string;  domain?: string;  instance?: string;  purchaseId: string;  purchaseLines: PurchaseLine[];}
```

Example

```
import  {useEffect} from 'react';import { View , Text} from "web" ;import Personalization from "@personalization-js-sdk/web"import { PurchaseData, EventTypes } from '@personalization-js-sdk/common';const PurchaseDataInitial : PurchaseData = {  account: "monetate",  domain: "localHost",  instance: "p",  purchaseId: "abc1c",  purchaseLines: [{    sku: "t-shirt",    pid: "0302",    quantity: "1",    currency: "USD",    value: "23",  }]  };const App = () => {     const PersonalizationInstance = new Personalization(account, user);     useEffect(() => {  PersonalizationInstance.report (    EventTypes.ContextPurchase, {...PurchaseDataInitial}  ),  }, []);       return <View>    <Text> Monetate SDK Purcahse Event.</Text>  </View>}export default App;
```

Closed Session Event

Imports ClosedSessionData from @personalization-js-sdk/common.

```
interface ClosedSessionData {  closedSession?: object;}
```

Example

```
import  {useEffect} from 'react';import { View , Text} from "web" ;import Personalization from "@personalization-js-sdk/web"import { closedSession, EventTypes } from '@personalization-js-sdk/common';const ClosedSessionDataInitial : ClosedSessionData = {  // You can add any key value pair within Closed Sessions as per your needs  closedSession: { account_id: 123,  has_cart: "t",   has_purchase: "f", ...}};const App = () => {     const PersonalizationInstance = new Personalization(account, user);     useEffect (() => {  PersonalizationInstance.report (    EventTypes.ContextClosedSession, {...ClosedSessionDataInitial}  ),  }, []);       return <View>    <Text>Monetate SDK Purcahse Event.</Text>  </View>}export default App;
```

Decision Event

This event does not need to be created. The SDK automatically creates a decision event when you use the getActions method and pass a requestID as a string.