

# iOS Event Code Samples

The following events are available to use.

## Metadata Event

### Example

```
import UIKitimport monetate_ios_sdk@mainclass AppDelegate: UIResponder,
UIApplicationDelegate { let requestId = "123456" var window: UIWindow? final var
objPersonalization = Personalization( account: Account( instance: "p", domain:
"localhost.org", name: "a-701b337c", shortname: "localhost"), user: User(deviceId:
"62bd2e2d-213d-463f-83bb-12c0b2530a14")) func application( _ application:
UIApplication, didFinishLaunchingWithOptions launchOptions:
[UIApplication.LaunchOptionsKey: Any]? ) -> Bool { metadataApi() return true } func
metadataApi() { let languageData = JSONValue(dictionaryLiteral: ("language", "en-US"))
let jsonVal = JSONValue(dictionaryLiteral: ("metadata", languageData))
objPersonalization.report(context: .Metadata, event: Metadata(metadata: jsonVal)) }}
```

## Screen Size Event

### Example

```
import UIKitimport monetate_ios_sdk@mainclass AppDelegate: UIResponder,
UIApplicationDelegate { let requestId = "123456" var window: UIWindow? final var
objPersonalization = Personalization( account: Account( instance: "p", domain:
"localhost.org", name: "a-701b337c", shortname: "localhost"), user: User(deviceId:
"62bd2e2d-213d-463f-83bb-12c0b2530a14")) func application( _ application:
UIApplication, didFinishLaunchingWithOptions launchOptions:
[UIApplication.LaunchOptionsKey: Any]? ) -> Bool { screenSizeApi() return true } func
screenSizeApi() { let (width, height) = MiscClass.getDeviceSize()
objPersonalization.report( context: .ScreenSize, event: ScreenSize(height:
Int(truncating: width), width: Int(truncating: height))) }}
```

## User Agent Event

### Example

```

import UIKitimport monetate_ios_sdk@mainclass AppDelegate: UIResponder,
UIApplicationDelegate { let requestId = "123456" var window: UIWindow? final var
objPersonalization = Personalization( account: Account( instance: "p", domain:
"localhost.org", name: "a-701b337c", shortname: "localhost"), user: User(deviceId:
"62bd2e2d-213d-463f-83bb-12c0b2530a14")) func application( _ application:
UIApplication, didFinishLaunchingWithOptions launchOptions:
[UIApplication.LaunchOptionsKey: Any]? ) -> Bool { userAgentApi() return true } func
userAgentApi() { let userAgent = WKWebView().value(forKey: "userAgent") as! String
objPersonalization.report(context: .UserAgent, event: UserAgent(userAgent: userAgent)) }}

```

## Coordinates Event

Reports the coordinates of the customer.

### Example

```

import UIKitimport monetate_ios_sdk@mainclass AppDelegate: UIResponder,
UIApplicationDelegate { let requestId = "123456" var window: UIWindow? final var
objPersonalization = Personalization( account: Account( instance: "p", domain:
"localhost.org", name: "a-701b337c", shortname: "localhost"), user: User(deviceId:
"62bd2e2d-213d-463f-83bb-12c0b2530a14")) func application( _ application:
UIApplication, didFinishLaunchingWithOptions launchOptions:
[UIApplication.LaunchOptionsKey: Any]? ) -> Bool { coordinateApi() return true } func
coordinateApi() { let coordinates = Coordinates(latitude: "11.2734", longitude: "45.2734")
objPersonalization.report(context: .Coordinates, event: coordinates) }}

```

## Referrer Event

Reports the referrer.

### Example

```

import UIKitimport monetate_ios_sdk@mainclass AppDelegate: UIResponder,
UIApplicationDelegate { let requestId = "123456" var window: UIWindow? final var
objPersonalization = Personalization( account: Account( instance: "p", domain:
"localhost.org", name: "a-701b337c", shortname: "localhost"), user: User(deviceId:
"62bd2e2d-213d-463f-83bb-12c0b2530a14")) func application( _ application:
UIApplication, didFinishLaunchingWithOptions launchOptions:
[UIApplication.LaunchOptionsKey: Any]? ) -> Bool { referrerApi() return true } func
referrerApi() { let referrer = Referrer(referrer: "www.monetate.com")
objPersonalization.report(context: .Referrer, event: referrer) }}

```

# Custom Variable Event

Reports custom variables.

## Example

```
import UIKitimport monetate_ios_sdk@mainclass AppDelegate: UIResponder,
UIApplicationDelegate { let requestId = "123456" var window: UIWindow? final var
objPersonalization = Personalization( account: Account( instance: "p", domain:
"localhost.org", name: "a-701b337c", shortname: "localhost"), user: User(deviceId:
"62bd2e2d-213d-463f-83bb-12c0b2530a14")) func application( _ application:
UIApplication, didFinishLaunchingWithOptions launchOptions:
[UIApplication.LaunchOptionsKey: Any]? ) -> Bool { customVariableApi() return true }
func customVariableApi() { let customVariable = CustomVariables(customVariables: [
CustomVariablesModel(variable: "favoriteTeam", value: "Blue Hens") ])
objPersonalization.report(context: .CustomVariables, event: customVariable) }}
```

# Page Event

Reports any events on the page.

## Example

```
import UIKitimport monetate_ios_sdkclass CategoryViewController: UIViewController { final
var objPersonalization = Personalization( account: Account( instance: "p", domain:
"localhost.org", name: "a-701b337c", shortname: "localhost"), user: User(deviceId:
"62bd2e2d-213d-463f-83bb-12c0b2530a14")) override func viewDidLoad() {
super.viewDidLoad() pageEvent() } func pageEvent() { let myEvent =
PageEvents(pageEvents: Set(["myEvent"])) objPersonalization.report(context: .PageEvents,
event: myEvent) }}
```

# Page View Event

Reports when a customer views a specified page.

## Example

```
import UIKitimport monetate_ios_sdkclass CategoryViewController: UIViewController { final
var objPersonalization = Personalization( account: Account( instance: "p", domain:
"localhost.org", name: "a-701b337c", shortname: "localhost"), user: User(deviceId:
"62bd2e2d-213d-463f-83bb-12c0b2530a14")) override func viewDidLoad() {
super.viewDidLoad() pageEvent() } func pageEvent() { let myEvent =
PageEvents(pageEvents: Set(["myEvent"])) objPersonalization.report(context: .PageEvents,
event: myEvent) }}
```

## Product Thumbnail Event

Reports when a customer views a product thumbnail.

### Example

```
import UIKitimport monetate_ios_sdkclass ProductListViewController: UIViewController { final
var objPersonalization = Personalization( account: Account( instance: "p", domain:
"localhost.org", name: "a-701b337c", shortname: "localhost"), user: User(deviceId:
"62bd2e2d-213d-463f-83bb-12c0b2530a14")) override func viewDidLoad() {
super.viewDidLoad() productThumbnailViewEvent() } func productThumbnailViewEvent()
{ let productsArr = [ "BackP_001", "BackP_002", "BackP_003", "BackP_004",
"BackP_005", "BackP_006", "BackP_007", "BackP_008", "BackP_009", "BackP_010", ]
objPersonalization.report( context: .ProductThumbnailView, event:
ProductThumbnailView(products: Set(productsArr))) }}
```

## Product Detail View Event

Reports when a customer views a product detail page.

### Example

```
import UIKitimport monetate_ios_sdkclass ProductDetailViewController: UIViewController {
final var objPersonalization = Personalization( account: Account( instance: "p", domain:
"localhost.org", name: "a-701b337c", shortname: "localhost"), user: User(deviceId:
"62bd2e2d-213d-463f-83bb-12c0b2530a14")) override func viewDidLoad() {
super.viewDidLoad() productDetailViewEvent() } func productDetailViewEvent() { let
products = [ Product(productId: "product72", sku: "product72-large-green"),
Product(productId: "product43", sku: "product43-medium-striped"), ]
objPersonalization.report( context: .ProductDetailView, event: ProductDetailView(products:
[products])) }}
```

## Recommendation Clicks Event

Reports recommendations click events.

## Example

```
import UIKitimport monetate_ios_sdkclass ProductDetailViewController: UIViewController {
    @IBOutlet weak var imageView: UIImageView! @IBOutlet weak var titleProduct: UILabel!
    @IBOutlet weak var price: UILabel! @IBOutlet weak var addToCartButton: UIButton!
    @IBOutlet weak var descriptionTextView: UITextView! @IBOutlet weak var recLabel: UILabel!
    @IBOutlet weak var recLineSeperatorView: UIView! @IBOutlet weak var collectionView:
    UICollectionView! var productObj: ProductObj? var recProductArr: [ProductObj] = [] var
    isRecProduct: Bool = false final var objPersonalization = Personalization( account: Account(
    instance: "p", domain: "localhost.org", name: "a-701b337c", shortname: "localhost"), user:
    User(deviceId: "62bd2e2d-213d-463f-83bb-12c0b2530a14")) override func viewDidLoad() {
    super.viewDidLoad() self.recLabel.isHidden = true self.recLineSeperatorView.isHidden =
    true self.collectionView.isHidden = true fetchRecommendations() imageView.image =
    productObj?.image titleProduct.text = productObj?.title price.text = "Price $"
    (String(productObj!.price)) collectionView.contentInset = UIEdgeInsets(top: 0, left: 10,
    bottom: 0, right: 10) descriptionTextView.textContainer.lineFragmentPadding = 0
    descriptionTextView.text = productObj?.description adjustUITextViewHeight(arg:
    descriptionTextView) } override func viewWillAppear(_ animated: Bool) { if
    self.isRecProduct { // Recording RecClicks objPersonalization.report( context:
    .RecClicks, event: RecClicks(recClicks: [(self.productObj?.recToken!)! as String])) } } func
    adjustUITextViewHeight(arg: UITextView) { arg.translatesAutoresizingMaskIntoConstraints
    = true arg.sizeToFit() arg.isScrollEnabled = false } func fetchRecommendations() {
    objPersonalization.addEvent( context: .PageView, event: PageView(pageType: "PDP",
    path: "n/a", url: "n/a", categories: [], breadcrumbs: [])) objPersonalization.getActionsData(
    requestId: "123456", includeReporting: false, arrActionTypes:
    ["monetate:action:OmnichannelRecommendation"] ).on { res in if res.status == 200 {
    self.handleRecommendations(res: res) } else { } } } fileprivate func
    handleRecommendations(res: APIResponse) { let data = JSON(res.data as Any) for item in
    data["data"]["responses"].arrayValue { if item["requestId"].string == res.requestId {
    for oneAction in item["actions"].arrayValue { if oneAction["component"].string ==
    "iOS_InApp_Recs" { recProductArr.removeAll() self.recLabel.isHidden = false
    self.recLineSeperatorView.isHidden = false self.collectionView.isHidden = false
    for productDict in oneAction["items"].arrayValue { for cat in
    CategoryGeneratorforProducts.categories { for prdct in cat.products { if
    prdct.pid == productDict["id"].rawValue as! String { prdct.recToken =
    productDict["recToken"].rawValue as? String prdct.recSetId =
    productDict["recSetId"].rawValue as? Double prdct.affinity =
    productDict["_affinity"].rawValue as? Double prdct.rawAffinity =
    productDict["_rawAffinity"].rawValue as? Double prdct.slotIndex =
    productDict["slotIndex"].rawValue as? Int recProductArr.append(prdct)
    } } } } } self.collectionView.reloadData()
    DispatchQueue.main.asyncAfter( deadline: .now() + 2.0, execute: {
    self.collectionView.reloadData() }) return } } } } } } @IBAction
    func addToCartButtonAction(_ sender: UIButton) { var cartline: [CartLine] = []
    cartline.append( CartLine( sku: productObj!.sku, pid: productObj!.pid, quantity:
```

```

productObj!.quantity, currency: "USD", value: String(productObj!.price *
Double(productObj!.quantity)))) isPageViewEvent = false isAddToCartEvent = true
}}extension ProductDetailViewController: UICollectionViewDelegate,
UICollectionViewDelegateFlowLayout, UICollectionViewDataSource{ func collectionView(
collectionView: UICollectionView, numberOfItemsInSection section: Int) -> Int { return
recProductArr.count } func collectionView(_ collectionView: UICollectionView, cellForItemAt
indexPath: IndexPath) -> UICollectionViewCell { let cell =
collectionView.dequeueReusableCell( withReuseIdentifier: "PDPCollectionViewCell", for:
indexPath) as! PDPCollectionViewCell let recProduct = recProductArr[indexPath.row]
cell.productImageView.image = recProduct.image cell.producttitleLabel.text =
recProduct.title cell.productPriceLabel.text = "$" + String(recProduct.price) return cell }
func collectionView( collectionView: UICollectionView, didSelectItemAt indexPath: IndexPath)
{ let controller = UIStoryboard(name: "Ecommerce", bundle:
nil).instantiateViewController( identifier: "ProductDetailViewController") as!
ProductDetailViewController controller.productObj = self.recProductArr[indexPath.row]
controller.isRecProduct = true self.navigationController?.pushViewController(controller,
animated: false) }}

```

## Recommendation Impressions Event

Reports recommendations impression events.

### Example

```

import UIKitimport monetate_ios_sdkclass ProductDetailViewController: UIViewController {
@IBOutlet weak var imageView: UIImageView! @IBOutlet weak var titleProduct: UILabel!
@IBOutlet weak var price: UILabel! @IBOutlet weak var addToCartButton: UIButton!
@IBOutlet weak var descriptionTextView: UITextView! @IBOutlet weak var recLabel: UILabel!
@IBOutlet weak var recLineSeperatorView: UIView! @IBOutlet weak var collectionView:
UICollectionView! var productObj: ProductObj? var recProductArr: [ProductObj] = [] var
isRecProduct: Bool = false final var objPersonalization = Personalization( account: Account(
instance: "p", domain: "localhost.org", name: "a-701b337c", shortname: "localhost"), user:
User(deviceld: "62bd2e2d-213d-463f-83bb-12c0b2530a14")) override func viewDidLoad() {
super.viewDidLoad() self.recLabel.isHidden = true self.recLineSeperatorView.isHidden =
true self.collectionView.isHidden = true fetchRecommendations() imageView.image =
productObj?.image titleProduct.text = productObj?.title price.text = "Price $\
(String(productObj!.price))" collectionView.contentInset = UIEdgeInsets(top: 0, left: 10,
bottom: 0, right: 10) descriptionTextView.textContainer.lineFragmentPadding = 0
descriptionTextView.text = productObj?.description adjustUITextViewHeight(arg:
descriptionTextView) } override func viewWillAppear(_ animated: Bool) { if
self.isRecProduct { // Recording ReImpressions objPersonalization.report( context:
.RecImpressions, event: RecImpressions(recImpressions: [(self.productObj?.recToken!)
as String]) ) } } func adjustUITextViewHeight(arg: UITextView) {
arg.translatesAutoresizingMaskIntoConstraints = true arg.sizeToFit() arg.isScrollEnabled
= false } func fetchRecommendations() { objPersonalization.addEvent( context:

```

```

.Pageview, event: Pageview(pageType: "PDP", path: "n/a", uri: "n/a", categories: [],
breadcrumbs: [])) objPersonalization.getActionsData( requestId: "123456",
includeReporting: false, arrActionTypes:
["monetate:action:OmnichannelRecommendation"] ).on { res in if res.status == 200 {
self.handleRecommendations(res: res) } else { } } } fileprivate func
handleRecommendations(res: APIResponse) { let data = JSON(res.data as Any) for item in
data["data"]["responses"].arrayValue { if item["requestId"].string == res.requestId {
for oneAction in item["actions"].arrayValue { if oneAction["component"].string ==
"iOS_InApp_Recs" { recProductArr.removeAll() self.recLabel.isHidden = false
self.recLineSeperatorView.isHidden = false self.collectionView.isHidden = false
for productDict in oneAction["items"].arrayValue { for cat in
CategoryGeneratorforProducts.categories { for prdct in cat.products { if
prdct.pid == productDict["id"].rawValue as! String { prdct.recToken =
productDict["recToken"].rawValue as? String prdct.recSetId =
productDict["recSetId"].rawValue as? Double prdct.affinity =
productDict["_affinity"].rawValue as? Double prdct.rawAffinity =
productDict["_rawAffinity"].rawValue as? Double prdct.slotIndex =
productDict["slotIndex"].rawValue as? Int recProductArr.append(prdct)
} } } } self.collectionView.reloadData()
DispatchQueue.main.asyncAfter( deadline: .now() + 2.0, execute: {
self.collectionView.reloadData() }) return } } } } } @IBAction
func addToCartButtonAction(_ sender: UIButton) { var cartline: [CartLine] = []
cartline.append( CartLine( sku: productObj!.sku, pid: productObj!.pid, quantity:
productObj!.quantity, currency: "USD", value: String(productObj!.price *
Double(productObj!.quantity))) isPageViewEvent = false isAddToCartEvent = true
})}extension ProductDetailViewController: UICollectionViewDelegate,
UICollectionViewDelegateFlowLayout, UICollectionViewDataSource{ func collectionView(_
collectionView: UICollectionView, numberOfItemsInSection section: Int) -> Int { return
recProductArr.count } func collectionView(_ collectionView: UICollectionView, cellForItemAt
indexPath: IndexPath) -> UICollectionViewCell { let cell =
collectionView.dequeueReusableCell( withReuseIdentifier: "PDPCollectionViewCell", for:
indexPath) as! PDPCollectionViewCell let recProduct = recProductArr[indexPath.row]
cell.productImageView.image = recProduct.image cell.producttitleLabel.text =
recProduct.title cell.productPriceLabel.text = "$" + String(recProduct.price) return cell }
func collectionView(_ collectionView: UICollectionView, didSelectItemAt indexPath: IndexPath)
{ let controller = UIStoryboard(name: "Ecommerce", bundle:
nil).instantiateViewController( identifier: "ProductDetailViewController") as!
ProductDetailViewController controller.productObj = self.recProductArr[indexPath.row]
controller.isRecProduct = true self.navigationController?.pushViewController(controller,
animated: false) }}

```

## Impressions Event

Reports impression events.

## Example

```
import UIKit
import monetate_ios_sdk
class ProductDetailViewController: UIViewController {
    final var objPersonalization = Personalization( account: Account( instance: "p", domain:
    "localhost.org", name: "a-701b337c", shortname: "localhost"), user: User(deviceId:
    "62bd2e2d-213d-463f-83bb-12c0b2530a14")) override func viewDidLoad() {
    super.viewDidLoad() recordImpressionsEvent() } func recordImpressionsEvent() { let
    impressionIds = ["3.MS4xLjE1MTQ4MDg5MDAuMDAwMDAw"] objPersonalization.report(
    context: .Impressions, event: Impressions(impressionIds: impressionIds)) }}
```

## 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.

## Example



```

import MarqueeLabelimport UIKitimport monetate_ios_sdkclass ProductDetailViewController:
UIViewController { @IBOutlet weak var imageView: UIImageView! @IBOutlet weak var
titleProduct: UILabel! @IBOutlet weak var price: UILabel! @IBOutlet weak var
addToCartButton: UIButton! @IBOutlet weak var descriptionTextView: UITextView! var
productObj: ProductObj? var quantityOfProduct = 0 var cartCount = 0 var lblBadge:
UILabel! var isAddToCartEvent: Bool = false final var objPersonalization = Personalization(
account: Account( instance: "p", domain: "localhost.org", name: "a-701b337c", shortname:
"localhost"), user: User(deviceId: "62bd2e2d-213d-463f-83bb-12c0b2530a14")) override
func viewDidLoad() { super.viewDidLoad() imageView.image = productObj?.image
titleProduct.text = productObj?.title price.text = "Price $\((String(productObj!.price))"
descriptionTextView.textContainer.lineFragmentPadding = 0 descriptionTextView.text =
productObj?.description adjustUITextViewHeight(arg: descriptionTextView) let cartBtn =
UIButton(type: .custom) cartBtn.frame = CGRect(x: 0, y: 0, width: 35, height: 35)
cartBtn.setImage(UIImage(named: "shopping-cart"), for: .normal) cartBtn.addTarget(self,
action: #selector(openCart), for: .touchUpInside) self.lblBadge = UILabel.init(frame:
CGRect.init(x: 20, y: 0, width: 15, height: 15)) self.lblBadge.backgroundColor =
MiscClass.hexStringToUIColor(hex: "E80101") self.lblBadge.clipsToBounds = true
self.lblBadge.layer.cornerRadius = 7 self.lblBadge.textColor = UIColor.white
self.lblBadge.font = UIFont(name: "Roboto-Regular", size: 10) self.lblBadge.textAlignment =
.center cartBtn.addSubview(self.lblBadge) self.navigationItem.rightBarButtonItem =
[UIBarButtonItem.init(customView: cartBtn)] addToCartButton.titleLabel?.font =
UIFont(name: "Roboto-Regular", size: 16) } override func viewWillAppear( _ animated: Bool)
{ updateCartIcon() DispatchQueue.main.asyncAfter( deadline: .now() + 4.0,
execute: { self.imageView.image = self.productObj?.image }) } func
adjustUITextViewHeight(arg: UITextView) { arg.translatesAutoresizingMaskIntoConstraints
= true arg.sizeToFit() arg.isScrollEnabled = false } func getCartCount() { var count = 0
for cat in CategoryGeneratorforProducts.categories { for prdct in cat.products { if
prdct.quantity > 0 { count = count + prdct.quantity } } } self.cartCount =
count } func updateCartIcon() { getCartCount() if cartCount > 0 { self.lblBadge.text
= "\((cartCount)" self.lblBadge.isHidden = false } else { self.lblBadge.isHidden = true
} } @objc func openCart() { let stroy = UIStoryboard(name: "Ecommerce", bundle: nil)
let controller = stroy.instantiateViewController(identifier: "EcomCartViewController")
as! EcomCartViewController self.navigationController?.pushViewController(controller,
animated: false) } @IBAction func addToCartButtonAction( _ sender: UIButton) {
productObj?.quantity += 1 updateCartIcon() var cartline: [CartLine] = []
cartline.append( CartLine( sku: productObj!.sku, pid: productObj!.pid, quantity:
productObj!.quantity, currency: "USD", value: String(productObj!.price *
Double(productObj!.quantity)))) isPageViewEvent = false isAddToCartEvent = true
objPersonalization.getActions( requestId: "123456", includeReporting: false,
arrActionTypes: ["monetate:action:Omnichanneljson"], eventsDict: [ .PageView:
PageView( pageType: "PDP", path: "n/a", url: "n/a", categories: [], breadcrumbs: []),
.AddToCart: AddToCart(cartLines: cartline), ] ).on { res in if res.status == 200 {
self.handleAction(res: res) } else { } } } }

```

# 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.

## Example

```
import JGProgressHUD import UIKit import monetate_ios_sdkclass
PurchaseProductViewController: UIViewController { var productforCart: [ProductObj] = []
@IBOutlet weak var topLabel: UILabel! @IBOutlet weak var horizontalCentreView: UIView!
@IBOutlet weak var purchaseBtn: UIButton! @IBOutlet weak var tableView: UITableView!
var spinner: SpinnerView? var isPurchaseSuccessful = false final var objPersonalization =
Personalization( account: Account( instance: "p", domain: "localhost.org", name: "a-
701b337c", shortname: "localhost"), user: User(deviceld: "62bd2e2d-213d-463f-83bb-
12c0b2530a14")) override func viewDidLoad() { super.viewDidLoad() tableView.delegate
= self tableView.dataSource = self tableView.separatorStyle = .none
horizontalCentreView.isHidden = true self.title = "Checkout" } override func
viewWillAppear(_ animated: Bool) { var totlaprice = 0.0 for prod in productforCart {
totlaprice += prod.price * Double(prod.quantity) } topLabel.text = "Total Price of
Products : ${totlaprice}" } @IBAction func goDashboard(_ sender: Any) {
self.navigationController?.popToViewController(
(self.navigationController?.viewControllers.first)!, animated: false) } @IBAction func buy(_
sender: Any) { let hud = JGProgressHUD() hud.textLabel.text = "Purchasing.."
hud.show(in: self.view) hud.dismiss(afterDelay: 3, animated: true) { [self] in let alert =
UIAlertController( title: "Payment", message: "Would you like to proceed with payment?",
preferredStyle: .alert ) alert.addAction( UIAlertAction( title: "Yes", style:
.default, handler: { action in let hud2 = JGProgressHUD()
hud2.textLabel.text = "Payment.." hud2.show(in: self.view)
hud2.dismiss(afterDelay: 3, animated: true) { [self] in placeOrderDidSelect()
print("Purchased") } }))) alert.addAction( UIAlertAction( title: "No",
style: .destructive, handler: { action in }))) self.present(alert, animated: true,
completion: nil) } } func placeOrderDidSelect() { var cartline: [PurchaseLine] = [] for
item in productforCart { cartline.append( PurchaseLine( sku: item.sku, pid:
item.pid, quantity: item.quantity, currency: "USD", value: String(Double(item.quantity) *
item.price))) } objPersonalization.addEvent( context: .PageView, event: PageView(
pageType: "Purchase", path: "n/a", url: "n/a", categories: [], breadcrumbs: []))
objPersonalization.addEvent( context: .Purchase, event: Purchase( account:
"Amazon", domain: "amazon.com", instance: "temp", purchaseld: "pur-23232",
purchaseLines: cartline)) objPersonalization.getActionsData( requestld: "123456",
includeReporting: false, arrActionTypes: ["monetate:action:Omnichanneljson"] ) .on {
(res) in self.isPurchaseSuccessful = true self.handleAction(res: res) } } fileprivate
func handleAction(res: APIResponse) { let data = JSON(res.data) for item in data["data"]
["responses"].arrayValue { if item["requestld"].string == res.requestld { for oneaction
in item["actions"].arrayValue { let component = oneaction["component"].string ?? ""
if component.lowercased() == "footer" { if let json = oneaction["json"].dictionary {
print("final dict \(json)") if let text = json["text"]?.string { if
isPurchaseSuccessful { for item in productforCart {
```

