Olo Serve Global Data and Events

TL;DR

Use this guide to integrate Google Tag Manager (GTM) with Olo Serve.

Introduction

When using a tag manager like GTM, it's useful to have direct access to application data instead of scraping the document for this information. Serve exposes a subset of its data globally so that a tag author can determine the state of application data at any time. It's also helpful to know when a user does *something*, like add or remove an item to their basket.

For example, a tag author may want to know how many items were in the user's basket when they add another to the cart. To do this, the tag could use custom Javascript to do something like:

```
window.Olo.on('v1.addToCart', function (prd) {
  doSomethingWithTheData({
    product: prd,
    itemsInCart: window.olo.data.basket.basketProducts.length,
  });
});
```

Setup

To use Serve's global events, simply reference `window.Olo.on` from inside of a `<script>` block in a GTM "Custom HTML" tag. From there, you can interact with the `dataLayer` directly, or run any kind of Javascript you need.

Here is an example of creating a custom GTM event when an item is added to the cart, and referencing some global data at the same time. You could use this event to, for example, trigger a push to Google Analytics from GTM.

```
<script>
  window.Olo.on('v1.addToCart', function (bp) {
    dataLayer.push({
      event: 'productAddedToCart',
      ecommerce: {
        currencyCode: window.Olo.data.vendor.currency,
        add: {
          products: [
            {
              name: bp.productName,
              id: bp.product.id,
              price: bp.unitCost,
              quantity: bp.quantity,
            },
          ],
        },
      },
    });
  });
</script>
```

For a simple integration with Google Analytics, see the official GTM template.

Developer Notes

When Olo Serve loads, global data and events are initialized *before* GTM. Because of this, there is no need to worry about checking if the `Olo` object exists on the `window` from inside your custom scripts.

API

Events

To know when a user has completed an action, Serve exposes a global event bus that will run provided callback functions. To listen for an event, use the `window.Olo.on` function, passing it an event name and a callback function as its two arguments. To stop listening to an event, use `window.Olo.off` and pass it the same event name and callback.

Here is a list of the available events and their corresponding callback payloads. See Models below for more details.

Event Name	Provided Callback Arguments	Description
`v1.addToCart`	`BasketProduct`	Run when a product is added to the cart.
`v1.removeFromCart`	`BasketProduct`	Run when a product is removed from the cart.
`v1.checkout`	`Basket, CallbackFn`	Run when the "Checkout" button is pressed from the cart. The `CallbackFn` must be run when the tracking is done to allow the app to transition to the Checkout page.
`v1.transaction`	`Order, OrderSubmission, User`	Run on the Thank You page when a transaction has completed.
`v1.clickProductLink`	`Product, 'cart-upsell' | 'category' | 'vendor-menu'`	Run when a product is clicked anywhere on the site. Second argument denotes which context the product was in when it was clicked.
`v1.viewProductDetail`	`Product, 'modal' | 'page'`	Run when loading the product details page.
`v1.productsVisible`	`Product[]`	Run when products become visible when viewing a menu page.

Event replay

To account for scenarios where GTM is initialized *after* Serve has fired an event, by default `Olo.on` will "replay" any events that occurred before GTM loaded. If you don't wish to run your provided callback for all previously fired events of a certain type pass a third parameter to `Olo.on` that looks like `{ replay: false }`.

Globals

Here is the data exposed globally by Serve for access by any GTM tag. See Models below for more details.

```
window.Olo.data = {
  vendor: Vendor;
  basket: Basket;
  product: Product;
  order: Order;
  user: User;
}
```

Models

`Basket`

```
{
  id: string;
  basketType: string;
  coupon: {
    code: string | number;
    description: string;
  };
  couponDiscount: number;
  deliveryMode:
    'None' |
    'Pickup' |
    'Curbside' |
    'Delivery' |
    'Dispatch' |
    'DriveThru' |
    'DineIn'
  fees: [{
    description: string;
    amount: number;
    name: string;
  }];
  feesTotal: number;
  guid: string;
  hasReward: boolean;
  isAdvance: boolean;
  isCurbside: boolean;
  isDelivery: boolean;
  isDriveThru: boolean;
  isGroup: boolean;
  isImmediate: boolean;
  isManualFire: boolean;
  isUpsellEnabled: boolean;
  leadTimeEstimate: number;
  reward: Reward;
  subTotal: number;
  suggestedTip: number;
  taxes: [{
    totalTax: number;
    label: string;
    rate: number;
  }];
  timeWanted: string;
  timeWantedIso: string;
  tip: number;
  total: number;
  vendorDiscount: number;
  wasUpsold: boolean;
  basketProducts: BasketProduct[];
}
```

BasketProduct

```
{
  productName: string;
  categoryName: string;
  recipientName: string;
  specialInstructions: string;
  quantity: number;
  menuViewId: number;
  totalCost: number;
  unitCost: number;
  customizeDescription: string;
  product: Product;
}
```

`Order`

```
{
  id: string;
  orderGuid: string;
  displayId: string;
  currency: string;
  currentTotal: number;
  deliveryCharge: number;
  deliveryMode:
    'N' | // Unspecified
    'P' | // CounterPickup
    'C' | // CurbsidePickup
    'D' | // Delivery
    'T' | // Dispatch
    'U' | // DriveThru
    'I' | // DineIn
  ;
  handoffDescription: string;
  isAdvance: boolean;
  isFavorite: boolean;
  manualFireHandoffInstructions: string;
  status:
    'Pending' |
    'New' |
    'FaxPending' |
    'Emailed' |
    'InProgress' |
    'Closed' |
    'Failed' |
    'Cancelled' |
    'Scheduled' |
  ;
  statusDescription: string;
  subTotal: number;
  thanksHandoffInstructions: string;
  tip: number;
  timePlacedLocal: string;
  timePlacedUtc: string;
  timeReadyLocal: string;
  totalCost: number;
  utcOffset: number;
  vendorDiscount: number;
  vendorName: string;
  vendorTax: number;
  vendorTotal: number;
  guestNewsletterOptIn: boolean;
  fees: [{
    description: string;
    amount: number;
    name: string;
  }];
  feesTotal: number;
  basketProducts: GlobalBasketProduct[];
  vendor: Vendor;
}
```

Note: this replaces `window.OLO.ecomTrackingObj` which was recommended previously.

`OrderSubmission`

```
{
  selectedBillingMethods,
  basket: {
    basketProducts: [{
      productName: string;
      quantity: number;
    }];
    coupon: {
      code: string | number;
      description: string;
    };
    deliveryCharge: number;
    deliveryMode:
      'None' |
      'Pickup' |
      'Curbside' |
      'Delivery' |
      'Dispatch' |
      'DriveThru' |
      'DineIn';
    guid: string;
    reward: Reward;
    subTotal: number;
    taxes: [{
      totalTax: number;
      label: string;
      rate: number;
    }];
    timeWantedType:
      1 | // Advance
      2 | // Immediate
      4; // ManualFire
    total: number;
    vendorDiscount: number;
    vendor: {
      address: string;
      externalReference: string;
      name: string;
    }
  };
  loyaltyAccount: [{
    membershipId: string;
    schemeProviderName: string;
  }];
  memberships: [{
    id: string;
    isGiftCard: boolean
  }];
};
```

Product

```
{
  id: string;
  name: string;
  description: string;
  shortDescription: string;
  baseCost: number;
  baseCostOverrideLabel: string;
  calorieLabel: string;
  maximumQuantity: number;
  minimumQuantity: number;
  isFeatured: boolean;
  images: [{
    groupName: string;
    filename: string;
  }];
  labels: [{
    code:
      'NYCHighSodium' |
      'PHLSodiumWarning' |
      'Prop65Toxic';
    name: string;
    icon: {
      name: string;
      alt: string;
      url: string;
      width: number;
      height: number;
    };
  }];
  availability: {
    startDate: string;
    endDate: string;
    weekTimes: [{
      start: {
        dayOfWeek: string;
        hour: number;
        minute: number;
      };
      end: {
        dayOfWeek: string;
        hour: number;
        minute: number;
      };
      spansAllDay: boolean;
      spansMultipleDays: boolean;
    }]
  };
  hasImages?: boolean;
  hasPrice?: boolean;
}
```

Reward

```
{
  id: number;
  externalReference: string;
  label: string;
  loyaltyRewardRule: {
    qualifyingProducts: number[];
    qualifyingCategories: number[];
    isDisabled: boolean;
    description: string;
    imageUrl: string;
    validMinutes: number;
    finePrint: string;
    availableOnline: boolean;
    availableOffline: boolean;
  };
}
```

User

```
{
  isLoggedIn: boolean;
  optIn: boolean;
  isEmbeddedLevelUp: boolean;
  isSsoLogin: boolean;
  isFacebook: boolean;
  currentCountry: 'us' | 'ca';
  uniqueId: string;
}
```

Vendor

```
{
  id: string;
  address: string;
  allowAdvanceOrders: boolean;
  allowCoupons: boolean;
  allowImmediateOrders: boolean;
  allowManualFireOrders: boolean;
  currency: string;
  displayNationalMenu: boolean;
  externalReference: string;
  isAcceptingOrders: boolean;
  isFavorite: boolean;
  latitude: number;
  longitude: number;
  minimumDeliveryOrder: number;
  minimumPickupOrder: number;
  name: string;
  phoneNumber: string;
  slug: string;
  status: 'Public' | 'Private';
  utcOffset: number;
}
```