[4.5.0] Geowidget Documentation (Old)

Geowidget Environments

1. Production environment available at: https://geowidget.easypack24.net/
2. Staging environment available at: https://sandbox-geowidget.easypack24.net/
3. Test environment available at: https://test-geowidget.easypack24.net/

Quick Start

SDK for JavaScript (SDK4JS) can be easily loaded into your HTML asynchronously, which means that it will not block loading of other components of your web site. You may put following code wherever you want, but if you want the map to load as soon as possible, put it at the top of the HTML <head> section - before the other scripts and stylesheets:

```
<script async src="https://sandbox-geowidget.easypack24.net/js/sdk-for-
javascript.js"></script>
<link rel="stylesheet" type="text/css" href="https://sandbox-geowidget.
easypack24.net/css/easypack.css">
```

And after that initialise widget by specifying DOM element ID (HTML) plus configuration;

```
<script type="text/javascript">
         window.easyPackAsyncInit = function () {
            easyPack.init({});
                var map = easyPack.mapWidget('easypack-map', function
(point) {
                        console.log(point);
                        alert(point.name, point.location_description);
                });
         };
 </script>
```

Last step is to add easypack-map placeholder and set it's height in part of your website/checkout where you want to display it:

```
<div style="width: 80%; height: 100%; margin: 20px auto;">
  <div id="easypack-map"></div>
</div>
```

IMPORTANT

Geowidget / SDK4JS will work on both Desktop and Mobile devices. Therefore placeholder where we embed widget should never be taller than mobile device screen height.

Basic Configuration

You may change default SDK for Javascript behaviour by passing configuration settings to easyPack.init() function:

Page contents

```
<script type="text/javascript">
         window.easyPackAsyncInit = function () {
            easyPack.init({
                    defaultLocale: 'uk',
                    closeTooltip: false,
                    points: {
                        types: ['parcel_locker']
                    },
                    map: {
                                defaultLocation: [51.507351, -0.127758]
                    }
                });
                var map = easyPack.mapWidget('easypack-map', function(point) {
                        console.log(point);
                        alert(point.name, point.location_description);
                });
         };
</script>
```

All configuration parameters

```
/*
  apiEndpoint adapts automatically based on 'defaultLocale' property. You can also
  manually configure it by overwriting URL in your local configuration.
 */
apiEndpoint: 'https://api-{locale}-points.easypack24.net/v1',
locales: ['pl'],  // options: pl, uk, ca, fr
defaultLocale: 'pl',
/*
  Server which serves javascripts, stylesheets and images.
 */
assetsServer: 'https://geowidget.easypack24.net',
infoboxLibraryUrl: '/js/lib/infobox.min.js',
markersUrl: '/images/desktop/markers/',
iconsUrl: '/images/desktop/icons/',
loadingIcon: '/images/desktop/icons/ajax-loader.gif',
closeTooltip: true, // close tooltip when details window is closed
mobileSize: 768,
points: {
  /*
   Type descriptions:
        parcel_locker - Parcel Lockers + PUDO Locations  / Paczkomaty +  Punkty PUDO,
        parcel_locker_only - Only Parcel Lockers / Wycznie Paczkomaty,
        laundry_locker - Laundry Lockers - Poland only / Pralniomaty - wycznie Polska,
        avizo_locker - Letter Locker - Poland only / Awizomaty - wycznie Polska
        pok - Customer Service Point / Punkt Obsugi Klienta,
        nfk - NFK Depo - Poland only / Oddzia NFK - wycznie Polska,
        avizo - InPost Awizo - Poland only,
        office - Office locations / Lokalizacje biur
   */
  types: ['parcel_locker', 'parcel_locker_only','laundry_locker', 'avizo_locker', 'pok', 'nfk', 'avizo',
'office'],
  fields: ['name', 'type', 'location', 'address'] // which point fields should be preloaded
},
map: {
  googleKey: xxx,  // Google Maps API key
  clusterer: {
    zoomOnClick: true,
    gridSize: 70,
    maxZoom: 12,
    minimumClusterSize: 3,
    styles: [
      {
        url: '/images/desktop/map-elements/cluster1.png',
```

```
        height: 61,
        width: 61
      },
      {
        url: '/images/desktop/map-elements/cluster2.png',
        height: 74,
        width: 74
      },
      {
        url: '/images/desktop/map-elements/cluster3.png',
        height: 90,
        width: 90
      }
    ]
  },
  useGeolocation: true,
  initialZoom: 13,
  detailsMinZoom: 15, // minimum zoom after marker click
  defaultLocation: [52.229807, 21.011595],
  /*
    Each time map center changes, MapWidget asks Points API for new points that are closest to map center.
    To improve loading speed and user experience, number of points returned by API is limited by two factors:
      1) Distance to relative point
         MapWidget calculates distance between map center and north east map bound.
         This is the distance that limits results to points that could be seen by user on map. But if the map
         is to be used with low density of points (for example only with parcel lockers) - you may want to
load
         more points than only those that will be visible. You can use distanceMultiplier to do that.
      2) Number of returned points
         If users zooms out the map, more and more points could be put on it. To keep the performance, there
is
         closestLimit setting which specifies how many points should be fetched at one request at max.
   */
  distanceMultiplier: 10,
  closestLimit: 200, // how many closest points should be loaded
  defaultDistance: 15, // distance for first request which may be called before map render
  initialTypes: ['pok', 'avizo', 'parcel_locker'], // which type should be selected by default
  reloadDelay: 250, // after how many milliseconds after bounds change should points be reloaded
  country: 'pl', // limits places library searches to particular country, set null to turn off
  typeSelectedIcon: '/images/desktop/icons/selected.png',
  typeSelectedRadio: '/images/mobile/radio.png',
  closeIcon: '/images/desktop/icons/close.png',
  pointIcon: '/images/desktop/icons/point.png',
  pointIconDark: '/images/desktop/icons/point-dark.png',
  detailsIcon: '/images/desktop/icons/info.png',
  pointerIcon: '/images/desktop/icons/pointer.png',
  tooltipPointerIcon: '/images/desktop/icons/half-pointer.png',
  photosUrl: '/uploads/{locale}/images/', // photos source based on current locale setting
  mapIcon: '/images/mobile/map.png',
  listIcon: '/images/mobile/list.png'
}
```

MapWidget

modalMap is responsible for displaying and manipulating map in modal window. This document describes how to use widget in modal window to return selected point data. See Quick Start for initialisation instructions and Basic Configuration for details how to configure.

> **IMPORTANT**
>
> mapWidget does work asynchronously. Once it's initialised in <head> section as advised in Quick Start it will start working without waiting for other website elements. Therefore if you want to bind following methods to any DOM elements, you should use window.onload event.

Point details callback

If you want to get data of selected point You can pass callback function as second parameter to easyPack.mapWidget. Widget automatically displays 'select' button inside tooltips. For example:

```
easyPack.mapWidget('map', function(point) {
    console.log(point);
});
```

showType()

Shows points of specified type. If mapWidget is in mobile mode, showing one type hides all other types.

Parameters

Name	Type	Description
type	string	One of points type: `'parcel_locker', 'laundry_locker', 'avizo_locker', 'pok', 'nfk', 'avizo`

Example: Show parcel lockers.

```
<script type="text/javascript">
    window.easyPackAsyncInit = function () {
        easyPack.init({});
        var map = easyPack.mapWidget('easypack-map');
        map.showType('parcel_locker');
    };
</script>
```

Example: Show parcel lockers on button click.

```
<script type="text/javascript">
    window.easyPackAsyncInit = function () {
        easyPack.init({});
        var map = easyPack.mapWidget('easypack-map');
        window.onload = function() {
            var button = document.getElementById('show-parcel-lockers');
            button.onclick = function() {
                map.showType('parcel_locker');
            }
        }
    };
</script>
```

hideType()

Hides points of specified type. This method has no effect if mapWidget is in mobile mode.

Parameters

Name	Type	Description
type	string	One of points type: `'parcel_locker', 'laundry_locker', 'avizo_locker', 'pok', 'nfk', 'avizo'`

Example: Hide avizo lockers

```
<script type="text/javascript">
  window.easyPackAsyncInit = function () {
    easyPack.init({});
    var map1 = easyPack.mapWidget('easypack-map');
    map1.hideType('avizo_locker');
  };
</script>
```

Example: Hide avizo lockers on button click

```
<script type="text/javascript">
    window.easyPackAsyncInit = function () {
        easyPack.init({});
        var map = easyPack.mapWidget('easypack-map');
        window.onload = function() {
            var button = document.getElementById('hide-avizo-lockers');
            button.onclick = function() {
                map.hideType('avizo_locker');
            }
        }
    };
</script>
```

searchPlace()

Triggers autocomplete service. This method has no effect if mapWidget is in mobile mode.

Parameters

Name	Type	Description
place	string	String used in Google Maps API. It could be street, city, country, coordinates

Example: Search for Cracow

```
<script type="text/javascript">
  window.easyPackAsyncInit = function () {
    easyPack.init({});
    var map1 = easyPack.mapWidget('easypack-map');
    map1.searchPlace('Kraków');
  };
</script>
```

Example: Search for Cracow after address change

```
<script type="text/javascript">
    window.easyPackAsyncInit = function () {
        easyPack.init({});
        var map1 = easyPack.mapWidget('easypack-map');

        var address = document.getElementById('address');
        address.onblur = function(){
            map1.searchPlace(address.value);
        };
    };
</script>
```

searchLockerPoint()

Searches for point with specified ID.

Parameters

Name	Type	Description
point	string	String containing to point ID (i.e. KRA334)

Example: Search for Cracow

```
<script type="text/javascript">
  window.easyPackAsyncInit = function () {
    easyPack.init({});
    var map1 = easyPack.mapWidget('easypack-map');
    map1.searchLockerPoint('KRA334');
  };
</script>
```

addType()

Dynamically add new type of points. This method has no effect if mapWidget is in mobile mode.

Parameters

Name	Type	Description
type	object	Object that contains id, name, description, icon, marker fields

Example: Add office type

```
<script type="text/javascript">
  window.easyPackAsyncInit = function () {
    easyPack.init({});
    var map1 = easyPack.mapWidget('easypack-map');
    map1.addType({
      id: 'office',
      name: 'Lokalizacje Biur',
      description: 'Biura w których mona nas znale',
      icon: 'https://geowidget.easypack24.net/images/desktop/icons/office.png',
      marker: 'https://geowidget.easypack24.net/images/desktop/markers/office.png'
    });
  };
</script>
```

addPoint()

Dynamically add new point of certain type. This method has no effect if mapWidget is in mobile mode.

Parameters

Name	Type	Description
point	object	Object that contains name, type, location, address

Example: Add new point in ul. Szkolna, Kraków

```
<script type="text/javascript">
  window.easyPackAsyncInit = function () {
    easyPack.init({});
    var map1 = easyPack.mapWidget('easypack-map');
    map1.addPoint({
      name: 'Biuro Szkolna',
      type: ['office'],
      location: {
        latitude: 50.023798,
        longitude: 19.965120
      },
      address: {
        line1: 'ul. Szkolna 13',
        line2: '30-624 Kraków'
      }
    });
  };
</script>
```

DropdownWidget

It is possible to display geowidget as dropdown with machines list. User can search machine by inputing it into search field:

```
easyPack.init({});
window.onload = function() {
    easyPack.dropdownWidget('easypack-widget', function(point) {
        console.log(point)
    });
}
```

Modal Map

modalMap is responsible for displaying and manipulating map in modal window. This document describes how to use widget in modal window to return selected point data. See Quick Start for initialisation instructions and Basic Configuration for details how to configure.

> **IMPORTANT**
>
> modalMap does work asynchronously. Once it's initialised in <head> section as advised in Quick Start it will start working without waiting for other website elements. Therefore if you want to bind following methods to any DOM elements, you should use window.onload event.

```
<script type="text/javascript">
window.easyPackAsyncInit = function () {
        easyPack.init({});
        window.onload = function() {
                var button = document.getElementById('popup-btn');
                button.onclick = function() {
                        easyPack.modalMap(function(point) {
                                this.close();   // Close modal with map, must be called from inside
modalMap() callback.
                                console.log(point);
                        }, {width: 500, height: 600 });
                }
        }
};
</script>
```

And in website body we include

```
<body>
  <a id="popup-btn"></a>
</body>
```

Points

points is part of Geowidget responsible for retrieval of InPost locations data. There are two methods that can be used as described below:

easyPack.points.closest()

Finds points closest to specified location. Maximum distance should be specified.

Parameters

Name	Type	Description
location	array	Array containing latitude and longitude.
distance	integer	Maximum distance from specified location to point in meters.
searchParams	object	Object containing machines search parameters.
callback	function	Function which you want to trigger as soon as machines are ready.

Example: Preload maximum 100 parcel_lockers closest to The Sopot Pier within the range of 10 kilometers and log them to console. Fetch only point name and location.

```
easyPack.points.closest([54.447012, 18.573502], 10000, { type: 'parcel_locker', fields: ['location'] },
function(points){ console.log(points) });
```

easyPack.points.find()

Finds machine by its id. It returns point object.

Parameters

Name	Type	Description
id	string	Id (name) of the point.
callback	function	Function which you want to trigger as soon as point is found

Example: Find machine with id "KRA302"

```
easyPack.points.find("KRA302", function(point){ console.log(point); });
```

Search parameters

Common set of parameters which may be applied to points methods.

Name	Type	Description
fields	enum{href, type, services, location, address, location_description, opening_hours}	Point fields which should be returned.
type	enum{parcel_locker, laundry_locker, avizo_locker, pok, nfk, avizo}	Filter by point type.