Betconstruct Sportsbook frontend integration document

version 0.27

Jun 02, 2016

Table of contents

Revision history

Name	Date	Version	Changes
Arsen Soghomonyan (arsen.soghomonyan@betconstruct.com)	Dec. 29, 2014	0.1	Initial version
Arsen Soghomonyan (arsen.soghomonyan@betconstruct.com)	Jul. 15, 2015	0.12	Added switchSportsbookLayout callback and new languages
Tigran Niksalyan (tigran.niksalyan@betconstruct.com)	Jul. 23, 2015	0.14	Added ‘userInactive’ callback option and new languages
Arsen Soghomonyan (arsen.soghomonyan@betconstruct.com)	Oct 07, 2015	0.15	Added ‘deposit’ button callback description
Arsen Soghomonyan (arsen.soghomonyan@betconstruct.com)	Nov 17, 2015	0.16	Added “Controlling sportsbook via window.postMessage” chapter
Tigran Niksalyan (tigran.niksalyan@betconstruct.com)	Dec 02, 2015	0.17	Added “favoriteGamesCount”
Tigran Niksalyan (tigran.niksalyan@betconstruct.com)	Dec 02, 2015	0.18	Added “logout”
Arsen Soghomonyan (arsen.soghomonyan@betconstruct.com)	Jan 25, 2016	0.2	Added “External Betslip” section
Tigran Niksalyan (tigran.niksalyan@betconstruct.com)	Feb 10, 2016	0.21	Added “routeUpdate”
Vahagn Grigoryan (vahagn.grigoryan@betconstruct.com)	Feb 11, 2016	0.22	Added “combo” view
Tigran Niksalyan (tigran.niksalyan@betconstruct.com)	Mar 29, 2016	0.23	Added “sportsBookLayout”
Tigran Niksalyan (tigran.niksalyan@betconstruct.com)	Apr 12, 2016	0.24	Added “doBet”
Arsen Soghomonyan (arsen.soghomonyan@betconstruct.com)	Apr 27, 2016	0.25	Added missing oddType parameter values, fixed some typos
Tigran Niksalyan (tigran.niksalyan@betconstruct.com)	Apr 29, 2016	0.26	Added “openBetHistory”
Tigran Niksalyan (tigran.niksalyan@betconstruct.com)	Jun 02, 2016	0.27	Addied “applicationReady”, “loggedIn”, “loggedOut”, “loginRestored”

Summary

This document is intended for BetConstruct Clients (Operators), who have their own site with existing integrations to other game providers and wish to integrate Betconstruct Sportsbook in an **iframe**.

Requirements

To be able to provide parent-iframe communication mechanism in a cross-browser way setting **document.domain** of parent and iframe windows to same value is done by integration script.
Thus the iframe and the parent window **must have the same top-level domain**.

Integration

To include the iframe on website 2 tags have to be placed on page where it should be displayed:

```
<div id="bcsportsbookcontainer"></div>
<script id="bcsportsbook"
src="http://sportsbook.example.com/js/partnerinit.js?containerID=bcsportsbookcontainer&callbackName=bettingCB&oddsType=decimal&type=prematch&UserId=123456&AuthToken=anonymous&lang=eng&sport=852&region=327693"></script>
```

This will automatically create an iframe with sportsbook inside the container div element.

the <script> element id cannot be changed, it must be "bcsportsbook". the container <div> element's id can be changed but must be passed to script as a parameter(see below).

URL parameters passed to script are:

mandatory:

- **containerID** - the id of html element where sportsbook will be inserted
- **UserId** - ID of the user in partner platform (for partner API backend)
- **AuthToken** - auth token generated for user (to pass to Partner API backend). Can be set empty or "anonymous" if yo don't want to login the user.

optional:

- **lang** - 3 letter language code (see appendix 1 for available values)
- **sportsBookLayout** - sportsbook layout type
  Possible values modern, classic, asian, eastern, combo
- **oddsType** - displayed odds format ("*decimal*", "*fractional*" or "*american*")
- **dynamicheight** - if set, iframe will change it's size according to its' content (on initial load and on parent windows resize).
  This is only needed for the Modern version of sportsbook.
  For the Classic version iframe can have fixed height, contents will fit.
  Here's sample css for iframe:
  ```
  #bcsportsbookcontainer { float: left;width: 100%;height: 700px;position: relative;overflow: hidden;}
  #bcsportsbookcontainer iframe {height: 800px;position: absolute;top: -100px;width: 100%;left: 0;}
  ```

- **callbackName** - name of callback function that will be called by sportsbook(see detailed description below)

NOTE: all parameter names are case sensitive

Deep linking

The following deep linking parameters can be passed to script as parameters to open specific sport/region/competition/game on sportsbook load:

type - 0 or "prematch" for pre-match, 1 or "live" for live
sport - sport id
region - region id
competition - competition id
game - game id

These ids can be obtained from odds feed or by opening sportsbook frame in a separate window, navigating to specific game and copying them from page URL.

page (works if not only sportsbook is enabled for partner**) -** the route of page to display on initial load (e.g. “/poolbetting”, “/news”, “/casino” etc.). If not set the default page(sportsbook) will be loaded.

Callback function

callback function should be available in parent page's global scope (i.e. inside window object).
It's called in the following cases:

1. On balance update with 2 parameters: `('balance', <Balance Object>).` Balance object is a JS object containig balance information (amount, currency, etc.)
2. When clicking 'sign in', 'register' or 'deposit' buttons inside sportsbook(if requested and enabled for partner). In this case 2 parameters will be passed to function:
   - when clicking sign in button: `('login', 'betslip')`
   - when clicking register button: `('register', 'betslip')`
   - when clicking deposit button: `('deposit, 'betslip')`
     second parameter, "betslip", indicates from where the button is clicked.
3. When a game is deeplinked and it's automatically scrolled into view, the game element offset will be passed to callback function: `('selectedGameOffset', <offsetObject>)`, where <*offsetObject*> is an object containing game element top and left offsets
4. if requested and enabled for partner: when iframe size changes, it will call function with 2 parameters: `('windowSize', <SizeObject>)`, where <*SizeObject*> is an object containing iframe width and height
5. if requested and enabled for partner: when sportsbook body height changes, it will call function with 2 parameters: `('bodyHeight', <height>)`, where <*height*> is the body element height
6. On sportsbook layout change(if several sportsbook layouts are enabled), function will be called with 2 parameters: `('switchSportsbookLayout', <type>)`, where <*type*> is the type that sportsbook is being switched to (‘modern’, ‘classic’ or ‘asian’)
7. if requested and enabled for partner: if user is inactive for some time(configurable) function will be called with `('userInactive')` parameter. You can perform logout or do whatever is needed in this case.
8. You will receive `('applicationReady')`once application is ready to receive events.
9. If the user has been logged in you will receive `('loggedIn')`.
10. If the user has been logged out you will receive `('loggedOut')`.
11. In case of session restore `('loginRestored')` will be called.

Accessing sportsbook functionality from parent window

the integration script will create an object called **htmlHelper** in it's window(window containing iframe) scope, which have the following methods for accessing sportsbook functionality after it’s loaded:

- `switchTo(page)` - page parameter should be 'live' or 'prematch'. Will switch to live or pre-match games page.
- `getSportsbookUrl` - will return iframe URL
- `setOddsType(type)` - will set odds display format. type parameter should have one of the following values: "*decimal*", "*fractional*" , "*american*", “*malay*”, “*indo*” or “*hongkong*”
- `open(page)` - will open top slider with specified page. *page* parameter can be "mygames" for favorite games or "history" for bet history page.

Controlling sportsbook via window.postMessage

Sportsbook has an event listener, which accepts commands sent by window.postMessage
This functionality applies not only to integration mode, it works also for white-label(standalone) mode.
Command is defined by “action” field of message. The following commands are available:

openSlider - opens the top slider, content is specified by **tab** property.

Example:
```
window.postMessage({action: “openSlider”,  tab: “signInForm”}, “*”);
```

possible values for **tab** parameter are:

- signInForm - the login form
- forgotPasswordForm - password recovery form
- registrationForm - registration form
- deposit - deposit page
- withdraw - withdraw page
- settings - opens one of the setting pages. The page is specified by additional **page** parameter.

  Example:
  ```
  window.postMessage({action: “openSlider”,  tab: “settings”, page: “changepasswords”}, “*”);
  ```

  Possible values for **page** are:

  - details - user profile details page
  - changepassword - change password page
  - preferences - user preferences page
  - verifyaccount - account verification page
  - selfexclusion - self-exclusion page
  - depositlimits - deposit limits page

Note: if requested page is available only when user is logged in, login form will be opened first and requested page will be opened after user logs in.

openHelp - opens help content (one of the CMS pages from the footer), page is specified by the **page** property, which defines the page slug in CMS.

Example:
```
window.postMessage({action: "openHelp", page: “privacy-policy”}, “*”);
```

login - logs the user in.

Example:
```
window.postMessage({action: "login", user_id: “123456”, auth_token: “9e132e8d4e55a3fb35caaa399e68913a26c80fbb11ac332da95fd1”}, “*”);
```

Reqiired parameters:

- **user_id** - user id
- **auth_token** - user authentication token

logout - logs user out

Example:
```
window.postMessage({action: "logout"}, “*”);
```

switchLayout - switches sportsbook layout (if several layouts are enabled)

Example: `window.postMessage({action: "switchLayout", layout: “classic”}, “*”);`

Layout is specified by **layout** parameter. Possible values are:

- classic
- modern
- asian
- eastern
- combo

showAlert - shows a modal dialog

Example: `window.postMessage({action: "showAlert", content: 'Hello!', title: "Please confirm", buttons: [{text: "Google", url: "http://www.google.com", target: “_blank”}]}, '*');`

Parameters:

- content - text/html content of dialog
- title - dialog window title
- buttons - (optional) array of buttons to be shown on dialog. Each button is an object with the following parameters:
    - text - text on button
    - url - button url
    - target - button link target

openBetHistory opens bet history slider

closeDialog - closes the open dialog window

Example: `window.postMessage({action: "closeDialog"}, “*”);`

doBet - add a bet to the betslip

Example: `window.postMessage({action: "doBet", bet: {event: event, market: market, game: game, oddType: oddType}}, “*”);`

External BetSlip

It is possible to integrate the Sportsbook without the betslip, and put the betslip(to be developed by partner) in the parent frame(partner side). This functionality is not enabled by default.

In this case when clicking on events inside sportsbook a message will be sent to parent frame indicating that event has to be added to betslip. Parent window will get the following message via `window.postMessage` call:

```
{action: <action>, data: <data>}
```

where

- **action** will have one of 2 values :
  - **add_bet** in case event has to be added to betslip,
  - **remove_bet** in case event was already added to betslip

- **data** is an object containing the event/market/game data with corresponding fields

In case when event is removed from betslip, parent frame needs to notify sportsbook to remove the highlighting of event. The following message has to be sent to the iframe via `window.postMessage`:

```
window.postMessage({action: "remove_bet", eventId: <event id>}, "*");
```

where <event id> is the id of event(integer).

Appendix 1 - available language codes

Code	Language
arm	Armenian
bgr	Bulgarian
bos	Bosnian
chi	Chinese
cze	Czech
dut	Dutch
eng	English
fas	Farsi
fre	French
geo	Georgian
ger	German
gre	Greek
hrv	Croatian
ita	Italian
jpn	Japanese
kor	Korean
lit	Lithuanian
nor	Norwegian
mac	Macedonian
pol	Polish
por	Portuguese
pt-br	Brazilian Portuguese
rus	Russian
slv	Slovenian
spa	Spanish
srp	Serbian
swe	Swedish
tur	Turkish
ukr	Ukrainian

Appendix 2 - sample code utilizing all described functionality

```
<html><body>
<script>
(function(){
        window.bettingCB = function bettingCB(name, value) {
                if (name === "balance") {
                        console.log("balance:", value);
                        // make sure to perform logout on balance error
                } else if (name === "login") {
                        console.log("login button clicked from ", value);
                } else if (name === "register") {
                        console.log("register button clicked from ", value)
                } else if (name === "selectedGameOffset") {
                        console.log("current displayed game has offset - x:",
                        value.left, "y: ", value.top);
                } else if (name === "windowSize") {
                        console.log("iframe size:", value.width + "x" + value.height);
                } else if (name === "bodyHeight") {
                        console.log("iframe body height changed to:", value);
                } else if (name === "switchSportsbookLayout") {
                        console.log("Sportsbook type changed to:", value);
                } else if (name === "userInactive" || name === "logout") {
                        console.log("User inactive or logged out. Perform logout.");
                } else if (name === "favoriteGamesCount") {
                        console.log("Update favorite games count.", value);
                } else if (name === "routeUpdate") {
                        // This must be enabled in the partner config by request
                        console.log("Route has been changed.", value);
                } else if (name === "applicationReady") {
                        // Application is ready to receive post messages
                        console.log("Application is ready to receive post messages);
                } else if (name === "loggedIn") {
                        // triggered once connected to backend
                        console.log("Logged in");
                } else if (name === "loggedOut") {
                        // triggered once disconnected from backend
                        console.log("Logged out");
                } else if (name === "loginRestored") {
                        // Triggered once connection to backend is restored
                        console.log("Login restored");
                }



})();
</script>
Open:
<a href="#" onclick="window.htmlHelper.open('mygames');">mygames</a>
<a href="#" onclick="window.htmlHelper.open('history');">history</a>
Switch to:
<a href="#" onclick="window.htmlHelper.switchTo('live');”>live</a>
<a href="#" onclick="window.htmlHelper.switchTo('prematch');”>prematch</a>
Set odds dislay type:
<a href="#" onclick="window.htmlHelper.setOddsType('decimal');”>decimal</a>
```

```
<a href="#" onclick="window.htmlHelper.setOddsType('fractional');”>fractional</a>
<a href="#" onclick="window.htmlHelper.setOddsType('american');”>american</a>
<a href="#" onclick="window.htmlHelper.setOddsType('malay');”>malay</a>
<a href="#" onclick="window.htmlHelper.setOddsType('indo');”>indo</a>
<a href="#" onclick="window.htmlHelper.setOddsType('hongkong');”>hongkong</a>

<div id="bcsportsbookcontainer"></div>
<script id="bcsportsbook"
src="http://sportsbook.example.com/js/partnerinit.js?containerID=bcsportsbookcontai
ner&callbackName=bettingCB&oddsType=decimal&type=prematch&AuthToken=anonymous&lang=
eng&sport=852&region=327693"></script>

</body></html>
```