Javascript

Pre requisites:

- nodejs - can be found at http://nodejs.org
- This document refers to the code found at https://github.com/betfair/API-NG-sample-code/tree/master/javascript.

JavaScript and API-NG

This page contains some code snippets of JavaScript interaction with API-NG. The example shows how to use JavaScript to construct and send requests to get list of markets, followed by list of runners for thechosen market and an attempt place a bet.

In the sample code we use the json-rpc and rescript protocols. All requests are sent and received using the JSON format. JavaScript uses node.js platform which allows us to run javascript from the command line. For more information see node.js home page at: http://nodejs.org/.

The application accepts session token and app key as command line arguments and runs in any linux environment that has nodejs installed.

How to run

In order to run this example, install nodejs which can be downloaded from here, set your environment path variable to point to the node executable and invoke a command:

```
/<path-to-your-nodejs-bin-directory>/node JsonRpcApiNgClient.js <your app key> <your session token>
```

Code Snippets

In order to construct the JSON-RPC request to API-NG we need to define mandatory headers that we send per request :

- appkey - your app key
- ssid - your session token

```
var options = {
    hostname: 'beta-api.betfair.com',
    port: 443,
    path: '/json-rpc',
    method: 'POST',
    headers: {
        'X-Application' : appkey,
        'Accept': 'application/json',
        'Content-type' : 'application/json',
        'X-Authentication' : ssid
        }
    }
```

Example POST request to get horse race id :

- jsonRequest - your constructed request object
- DEFAULT_ENCODING - set to "utf-8"

```
var requestFilters = '{"filter":{}}';
var jsonRequest = constructJsonRpcRequest('listEventTypes', requestFilters );
var req = https.request(options,function (res){
            res.setEncoding(DEFAULT_ENCODING);
            res.on('data', function (chunk) {
                str += chunk;
            });

            res.on('end', function (chunk) {
                // On resposne parse Json and check for errors
                var response = JSON.parse(str);
                handleError(response);
                // Retrieve id from response and get next available horse race
                getNextAvailableHorseRace(options, response);
            });

        });
        // Send Json request object
        req.write(jsonRequest, DEFAULT_ENCODING);
        req.end();

        req.on('error', function(e) {
            console.log('Problem with request: ' + e.message);
            return;
        });
    }
```

Example POST request to get next available horse races :

- eventId - retrieved form previous request
- jsonDate - is a current date

```
var requestFilters = '{"filter":{"eventTypeIds": [' + eventId + '],"marketCountries":["GB"],"marketTypeCodes":
["WIN"],"marketStartTime":{"from":"'+jsonDate+'"}},"sort":"FIRST_TO_START","maxResults":"1","marketProjection":
["RUNNER_DESCRIPTION"]}}';
var jsonRequest = constructJsonRpcRequest('listMarketCatalogue', requestFilters );
var req = https.request(options,function (res){
            res.setEncoding('utf8');
            res.on('data', function (chunk) {
                str += chunk;
            });

            res.on('end', function (chunk) {
                var response = JSON.parse(str);
                handleError(response);
                // Get list of runners that are available in that race
                getListOfRunners(options, response);
            });
        });
        req.write(jsonRequest, 'utf-8');
        req.end();
        req.on('error', function(e) {
            console.log('Problem with request: ' + e.message);
            return;
        });
```

Example POST request to get list of runners in the market :

- marketId - specify a market that we want to get runners from

```
var requestFilters = '{"marketIds":["' + marketId + '"],"priceProjection":{"priceData":["EX_BEST_OFFERS"],"
exBestOfferOverRides":{"bestPricesDepth":2,"rollupModel":"STAKE","rollupLimit":20},"virtualise":false,"
rolloverStakes":false},"orderProjection":"ALL","matchProjection":"ROLLED_UP_BY_PRICE"}}';
var jsonRequest = constructJsonRpcRequest('listMarketBook', requestFilters );
var str = '';
var req = https.request(options,function (res){
            res.setEncoding(DEFAULT_ENCODING);
            res.on('data', function (chunk) {
                str += chunk;
            });

            res.on('end', function (chunk) {
                var response = JSON.parse(str);
                handleError(response);
                // Place bet on first runner
                placeBet(options, response, marketId);
            });
        });
        req.write(jsonRequest, DEFAULT_ENCODING);
        req.end();
        req.on('error', function(e) {
            console.log('Problem with request: ' + e.message);
            return;
        });
```

Example POST request to place a bet on random runner :

- selectionId - runner that we want to place a bet on
- customerRef - unique reference for that transaction
- size - the size of the bet
- price - the price of the bet

```
var requestFilters = '{"marketId":"'+ marketId+'","instructions":[{"selectionId":"' + selectionId + '","
handicap":"0","side":"BACK","orderType":"LIMIT","limitOrder":{"size":"' + size + '","price":"' + price + '","
persistenceType":"LAPSE"}}],"customerRef":"'+customerRef+'"}}';
var jsonRequest = constructJsonRpcRequest('placeOrders', requestFilters );
var req = https.request(options,function (res){
            res.setEncoding(DEFAULT_ENCODING);
            res.on('data', function (chunk) {
                str += chunk;
            });
            res.on('end', function (chunk) {
                var response = JSON.parse(str);
                handleError(response);
                console.log(JSON.stringify(response, null, DEFAULT_JSON_FORMAT));
            });
        });
        req.write(jsonRequest, DEFAULT_ENCODING);
        req.end();
        req.on('error', function(e) {
            console.log('Problem with request: ' + e.message);
            return;
        });
```