D2Api

D2Api rozhranie predstavuje základný komunikaný kanál so systémom D2000 pre webové aplikácie. Toto komunikané rozhranie je na strane klienta implementované v JavaScript-e a je postavené na knižnici CometD.

Nadviazanie a rozpojenie spojenia s D2000

Nadviazanie spojenia s D2000 sa realizuje vytvorením inštancie triedy *D2Api*, ktorej sa odovzdá konfigurácia pre CometD a následným zavolaním metódy *connect*. Parametre konfiguraného objektu pre CometD sú popísané v dokumentácii konfigurácie CometD.

Jediný povinný atribút je atribút *url*, ktorý uruje cestu ku CometD servletu na serveri - môže by zadaný ako absolútna cesta ("http://server:port/aplikacia/api/cometd") alebo ako relatívna cesta v rámci webového servera ("/aplikacia/api/cometd").

Medzi alšie užitoné atribúty patrí najmä atribút *maxNetworkDelay*, ktorý uruje dobu v milisekundách, po uplynutí ktorej je požiadavka na server považovaná za zlyhanú. Tento parameter je vhodné zvýši z predvolených 10 sekúnd na vyššie íslo, ak bežné, synchrónne volané RPC procedúry majú dobu spracovávania dlhšiu.

Volitene je možné definova obsluhu chybových stavov metódou *registerErrorHandler* - v rámci SmartWeb Frameworku je pre tento úel dodávaná funkcia *reportError*, ktorá v prípade chyby zobrazuje lokalizovaný dialóg.

```
import D2Api from "framework\d2\d2Api"
import reportError from "framework\basic\reportError"


let cometdConfiguration = {
    url: "/smartWeb/api/cometd",
    maxNetworkDelay: 60000
}
let d2Api = new D2Api(cometdConfiguration);
d2Api.registerErrorHandler(reportError);
d2Api.connect();
```

Na rozpojenie spojenia s D2000 slúži metóda *disconnect*.

```
d2Api.disconnect();
```

Poúvanie na zmeny hodnôt objektov D2000

Prihlási sa na odoberanie hodnôt objektov D2000 je možné metódou *subscribeObject*. Tejto metóde sa cez parametre odovzdá meno D2000 objektu, ktorého zmeny sa budú sledova a obslužná funkcia, ktorá sa pri zmene hodnoty objektu zavolá. Volitene ako 3 parameter je možné preda objekt s atribútmi *returnFields*, prípadne *returnTransformation*, ktoré uria, ktoré atribúty hodnoty majú by vrátené a prípadne transformáciu, ktorá sa má nad hodnotami vykona. Možné hodnoty atribútov *returnFields* a *returnTransformation* sú popísané v sekcii Serializácia dát medzi klientom a API rozhraniami.

```
// Obslužná funkcia zmeny hodnoty objektu
function onSecChange(subscription) {
    // objekt subscription.uniVal bude obsahova hodnotu a atribúty D2000 objektu
}
...
// Registrácia odberu hodnôt
d2Api.subscribeObject("Sec", onSecChange, {returnFields: ["FormattedValue"]});
```

Odber hodnôt objektu je možné zruši volaním metódy *unSubscribeObject*, kde sa ako parameter odovzdá zaregistrovaná obslužná procedúra alebo komponent SmartWeb Frameworku.

```
d2Api.unSubscribeObject(onSecChange);
```

Naítanie hodnôt archívnych objektov D2000

Naítanie hodnôt archívnych objektov D2000 je možné pomocou metód *loadArchive* a *loadArchives*, ktoré naítajú hodnoty jedného, resp. viacerých archívnych objektov v danom intervale. Funkcie vracajú Promise objekt, ktorý v prípade úspechu bude obsahova kompletné naítané dáta. Na volanie týchto funkcií je možné použi aj async/await syntax.

```
var extParameters = {};
var receiveDataStreamHandler = null;


// Naíta hodnoty objektu H.ArchivedValue v intervale <startDate, endDate>
var arcData = await d2Api.loadArchive("H.ArchivedValue", startDate, endDate, extParameters,
receiveDataStreamHandler).call();

// Naíta hodnoty objektov H.ArchivedValue1, H.ArchivedValue2 v intervale <startDate, endDate>
var arcDataMulti = await d2Api.loadArchives(["H.ArchivedValue1", "H.ArchivedValue2"], startDate, endDate,
extParameters).call();
```

Volitený parameter *extParameters* slúži na definovanie atribútov hodnôt, ktoré majú by vrátené - zoznam sa definuje v atribúte *returnFields* - popísaný v sekcii Serializácia dát medzi klientom a API rozhraniami. Okrem toho môže by parametrom *extParameters* zadefinované prevzorkovanie hodnôt pomocou atribútu *oversampleSeconds* (krok prevzorkovania je v sekundách) a taktiež je možné obmedzi poet vrátených hodnôt z archívu nastavením atribútu *limitDataLength*. V prípade vynechania parametra *extParameters* budú vrátené len hodnoty bez alších atribútov a neprebehne žiadne prevzorkovanie ani orezanie dát.

```
var extParameters = {
    returnFields: ["ValueTime"], // okrem hodnôt sú požadované aj asové znaky hodnôt
    oversampleSeconds: 3600,     // hodnoty budú prevzorkované s hodinovým krokom
    limitDataLength: 1000        // maximálne bude vrátených 1000 hodnôt
}
```

Pri naítaní väšieho množstva dát z archívu, sú tieto dáta zo systému D2000 priebežne streamované po dávkach. Ak je žiadané zachytáva a spracováva tieto iastkové údaje, tak je to možné pomocou voliteného parametra *receiveDataStreamHandler*, ktorý definuje funkciu s dvoma parametrami - v prvom budú odovzdané hodnoty dávky a v druhom je príznak i ide o poslednú dávku.

receiveDataStreamHandler

```
function receiveDataStreamHandler(data, noModeData) {
    // spracovanie dát
}
```

Volanie D2000 RPC implementovanej v ESL

Rovnako ako REST API aj D2Api umožuje volanie D2000 RPC procedúr napísaných v ESL alebo v Jave. Slúžia na to metódy *rpc* - volanie ESL RPC, *rpcJava* - volanie Java RPC a *rpcSBA* - volanie Java RPC na prenos binárnych dát. Prvé dva parametre týchto metód urujú meno event objektu (skriptu) a názov RPC procedúry. Všetky alšie parametre sú odovzdané ako parametre do volanej RPC procedúry. Štruktúra parametrov pre volania RPC je popísaná v asti Serializácia dát medzi klientom a API rozhraniami. Metódy vracajú objekt s metódou *call*, ktorá vykoná volanie RPC. Návratová hodnota volania RPC je Promise objekt, ktorý v prípade úspešného volania obsahuje požadované výstupné parametre RPC procedúry ako svoje atribúty vo forme Unival hodnôt.

```
// Zavolá RPC procedúru Sum na Event skripte E.SmartWebTutorial
const rpcResponse = await d2Api.rpc(
    "E.SmartWebTutorial",
    "Sum",
    1, // prvý parameter - vstupný
    2, // druhý parameter - vstupný
    {type: "real", returnAs: "sum"} // tretí parameter - výstupný
).call();
// rpcResponse.sum obsahuje výstupný parameter RPC ako UniVal hodnotu
```

Volaná RPC procedúra v ESL má nasledovný predpis:

```
RPC PROCEDURE Sum (IN REAL _a, IN REAL _b, REAL _c)
  _c := _a + _b
END Sum
```

alšie spôsoby volania D2000 RPC

D2Api umožuje okrem ESL RPC metód vola aj Java RPC metódy prostredníctvom mena eventu alebo mena procesu, zárove je možné špecifikova pri volaní aj meno interfacu. Všetky spôsoby volania RPC sú zhrnuté nižšie.

```
// Spôsoby volania ESL RPC cez meno eventu
d2Api.rpc(<meno eventu>, <meno RPC>, ...parametre).call(<timeout>);
d2Api.rpcWithInterface(<meno eventu>, <meno interfacu>, <meno RPC>, ...parametre).call(<timeout>);

// Spôsoby volania Java RPC cez meno eventu
d2Api.rpcJava(<meno eventu>, <meno RPC>, ...parametre).call(<timeout>);
d2Api.rpcJavaWithInterface(<meno eventu>, <meno interfacu>, <meno RPC>, ...parametre).call(<timeout>);

// Spôsoby volania RPC cez meno procesu
d2Api.rpcJapi(<meno procesu>, <meno RPC>, ...parametre).call(<timeout>);
d2Api.rpcJapiWithInterface(<meno procesu>, <meno interfacu>, <meno RPC>, ...parametre).call(<timeout>);

// Spôsob volania Java SBA RPC cez meno eventu s jedinym parametrom typu https://developer.mozilla.org/en-US
/docs/Web/JavaScript/Reference/Global_Objects/ArrayBuffer
d2Api.rpcJava(<meno eventu>, <meno RPC>, <array buffer parameter>).call(<timeout>);
```

Volanie JavaScript funkcií z D2000

D2Api umožuje aj spätné volania z D2000 na webový prehliada a to vo forme registrovaných JavaScript funkcií. Takéto volania sú užitoné najmä pre notifikácie rôznych udalostí, ktoré v systéme D2000 vznikajú asynchrónne. Na registráciu JavaScript funkcie, ktorú je možné vola z D2000 slúži metóda *subscribeRpc*. Jej parametrami sú názov RPC procedúry, pod ktorým bude funkcia dostupná v rámci systému D2000 a samotná JavaScript funkcia, ktorá bude pri spätnom volaní zavolaná. Všetky parametre funkcie sú objekty typu Unival. Na odregistrovanie funkcie spätného volania slúži metóda *unSubscribeRpc* s jediným parametrom, a to danou JavaScript funkciou.

```
// Registrovatená Javascript funkcia
smartWebSessionRPC(message) {
    // ...
}

// Registrácia funkcie
d2Api.subscribeRpc("SmartWebSessionRPC", smartWebSessionRPC);

// Odregistrovanie funkcie
d2Api.subscribeRpc(smartWebSessionRPC);
```

Aby bola funkcia volatená zo systému D2000, musí by známa session, na ktorej má by zavolaná. Toto je možné zisti volaním ubovolnej, inicializanej RPC procedúry, v rámci ktorej sa identifikuje proces, z ktorého bola zavolaná.

```
INT _caller ; HOBJ SmartWeb session, ktorá si vyžiadala spätné volania

RPC PROCEDURE InitCallbacks
  _caller := %GetRPCCallerProcess()
END InitCallbacks
```

Volanie registrovanej JavaScript funkcie z ESL potom vyzerá nasledovne:

```
CALL [(0)] SmartWebSessionRPC("Hello SmartWeb") ASYNC ON (_caller)
```

Pretože jedna SmartWeb session môže ma pod jedným názvom RPC procedúry zaregistrovaných viacero Javascript funkcií, ide tu prakticky o viacnásobné (multicast) volanie, a preto v om nie je možné použi výstupné parametre. Volanie zárove musí by asynchrónne.

Zmena hesla prihláseného užívatea

Rozhranie D2Api umožuje taktiež zmeni heslo prihláseného užívatea. Slúži na to metóda *changePassword*, ktorá má dva parametre - staré heslo a nové heslo a vracia Promise objekt.

```
let oldPassword = 'secret';
let newPassword = 's3Cr3t*';
d2Api.changePassword(oldPassword, newPassword).
    .then(onFulfilled => {
        if (onFulfilled.data === 'ok') {
            // password changed
        }
    });
```