BSN-DDC 基础网络
DDC SDK 详细设计

V1.3

北京红枣科技有限公司

2021 年 12 月

修改记录

日期	版本	修改说明	修改者
2021.12.9	V1.0	版本初始化	
2021.12.11	V1.0	1. 补充了调用时序图 2. 添加了交易查询、区块查询章节	
2021.12.15	V1.1	1. 添加了签名事件和数据解析章节 2. 细化了部分方法的核心逻辑说明	
2021.12.27	V1.1	1. 方法的调用者进行了阶定 2. 1155 的销毁和批量销毁调用者、入参以及核心逻辑进行了更新	
2022.1.6	V1.1	1. 附录针对武汉链的区块信息示例、交易回执信息示例以及交易信息示例进行了补充	
2022.1.10	V1.2	1. 721 添加安全生成方法 2. 1155 生成和批量生成添加附加数据参数 3. 数据解析部分 721 添加了安全生成的部分	
2022.1.11	V1.2	1. 更新 1155 批量生成和批量查询对应的 Map 类型	
2022.1.11	V1.2	1. 3.2.1.1 和 3.2.1.2 更新; 2. 3.2.2.6 和 3.2.2.7 更新; 3. 3.2.8.1.1 更新;	
2022.1.14	V1.2	1. 权限管理、费用管理、721 以及 1155 所有方法添加 sender 参数 (注：此参数用于签名事件中传递给链服务或业务系统，由业务系统针对所传的 sender 指定账户进行交易	

		签名)	
2022.1.17	V1.2	1. 1155 生成和批量生成定义成了安全生成和批量安全生成，方法名也进行了调整。 2. 2.3 节添加了接入 KEY 的参数说明 3. 添加了 4.4 章节-离线生成账户	
2022.1.19	V1.3	1. BSN-DDC-权限管理添加跨平台授权。 2. 721 和 1155 添加 URI 设置方法。 3. 数据解析添加对跨平台授权以及 721 和 1155 对应的 URL 设置数据解析处理。 4. 1155 的销毁和批量销毁添加 DDC 授权者也可以调用，以及 1155 的 URI 设置添加 DDC 的授权者也可以调用，数据解析部分，针对 URI 设置的事件解析添加了一个字段。	
2022.1.20	V1.3	1. DDC 业务费扣除事件通知添加 ddcId 字段。	
2022.1.21	V1.3	1. 添加 4.5 接入 URL 设置方法和 4.6 接入 Key 设置方法。	

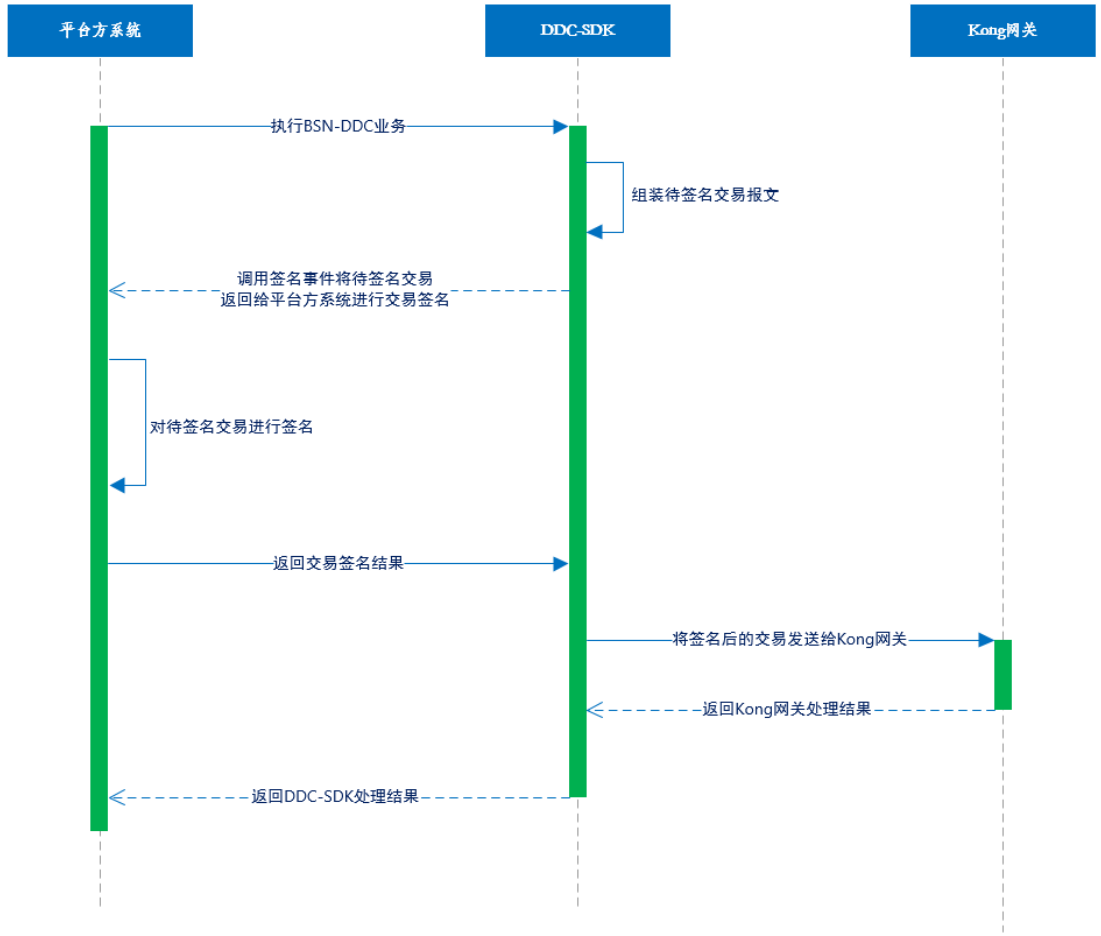
目录

1. 编写目的

为了让运营方或各平台方对 DDC-SDK 整体设计有一个全面详细的了解，同时为项目的开发、测试、验证、交付等环节提供原始依据以及开发指导，特此整理 DDC-SDK 整体设计规范方案说明文档。

2. 整体设计

2.1 调用时序图



2.2 开发语言标准

目前使用 Java 语言开发 SDK。

2.3 参数格式标准

- 时间

  格式为 yyyy-MM-dd HH:mm:ss 形式的字符串，例如: 2021-05-25 12:30:59 表示 2021 年 5 月 25 日 12 时 30 分 59 秒。

- 返回异常

当 SDK 处理功能逻辑出错时，会抛出相应的运行时异常，包含具体的错误信息。

- 接入 Key

如果在 DDC 门户创建项目时启用了项目 KEY 值，则在做 DDC 业务的时候，请求网关时需要附加上 KEY 值参数，此参数通过 Header 进行传递(注：KEY 在设置的时候 KEY 用“x-api-key”，值根据实际情况填写)。

3. 功能设计

3.1 DDC

所有的 BSN-DDC 方法都需要调用签名事件对待签名交易进行签名，签名事件必须由业务调用者服务进行注册并实现交易签名的业务逻辑。

3.1.1 BSN-DDC-权限管理

对账户进行管理，包含了账户的查询及更新账户状态的操作。

3.1.1.1 查询账户

3.1.1.3.1 功能介绍

运营方、平台方以及终端用户可以通过调用该方法进行 DDC 账户信息的查询。

3.1.1.3.2 API 定义

- 方法定义：AccountInfo getAccount(String account);
- 合约方法：getAccount(address account) returns (AccountInfo);
- 调用者：运营方、平台方以及终端用户;

- 核心逻辑:

  1. 检查 account 为标准 address 格式，并且不能为空地址;
  2. 检查签名事件是否被注册;

- 输入参数:

字段名	字段	类型	必传	备注
账户地址	account	String	是	DDC 用户链账户地址

- 输出参数:

字段名	字段	类型	必传	备注
账户 DID	accountDID	String	否	DDC 账户对应的 DID 信息（普通用户可为空）
账户名称	accountName	String	是	DDC 账户对应的账户名称
账户角色	accountRole	enum	是	DDC 账户对应的身份信息。值包含: 0. Operator（运营方） 1.PlatformManager（平台方） 2.Consumer（用户方）
账户上级管理者	leaderDID	String	是	DDC 账户对应的上级管理员，账户角色为 Consumer 时必填。对于普通用户 Consumer 该值为平台管理者 PlatformManager
平台管理账户状态	platformState	enum	是	DDC 账户对应的当前账户状态（仅平台方可操作该状态）。值包含:

				0.Frozen（冻结状态，无法进行 DDC 相关操作） 1.Active （活跃状态，可进行 DDC 相关操作）
运营管理账户状态	operatorState	enum	是	DDC 账户对应的当前账户状态（仅运营方可操作该状态）。值包含: 0.Frozen （冻结状态，无法进行 DDC 相关操作） 1.Active （活跃状态，可进行 DDC 相关操作）
冗余字段	field	String	否	冗余字段

3.1.1.2 更新账户状态

3.1.1.4.1 功能介绍

运营方或平台方可以通过调用该方法对终端用户进行 DDC 账户信息状态的更改。

3.1.1.4.2 API 定义

- 方法定义：String updateAccState(String sender,String account,State state,bool changePlatformState);
- 合约方法：updateAccountState(address account,State state,bool changePlatformState) returns (bool);
- 调用者：运营方、平台方;

- 核心逻辑:

  1. 检查 sender 为标准备 address 格式;
  2. 检查 account 为标准 address 格式，并且不能为空地址;
  3. 检查签名事件是否被注册;

- 输入参数:

字段名	字段	类型	必传	备注
调用者	sender	String	是	调用者地址
账户地址	account	String	是	DDC 用户链账户地址
状态	state	enum	是	1.Frozen（冻结状态，无法进行 DDC 相关操作） 2.Active （活跃状态，可进行 DDC 相关操作）
修改平台方状态标识	changePlatformState	Boolean	否	仅运营方调用有效。 当调用者为运营方且该字段为 true 时，运营方可修改某一账户的 platformState 为 state

- 输出参数:

字段名	字段	类型	必传	备注
		String	是	交易哈希

3.1.2 BSN-DDC-费用管理

3.1.2.1 充值

3.1.2.1.1 功能介绍

运营方或平台方可以通过调用该方法为所属同一方的同一级别账户或者下级账户进行充值;

3.1.2.1.2 API 定义

- 方法定义: String recharge(String sender,String to,BigInteger amount);
- 合约方法: recharge(address to,uint256 value) returns (bool success);
- 调用者: 运营方、平台方;
- 核心逻辑:
  1. 检查 sender 为标准备 address 格式;
  2. 检查 to 是有效的合约地址的哈希格式字符串，并且不是空地址;
  3. 检查 amount 必须大于零;
  4. 检查签名事件是否被注册;
- 输入参数:

字段名	字段	类型	必传	备注
调用者	sender	String	是	调用者地址
用户账户地址	to	String	是	充值账户的地址
金额	amount	BigInteger	是	转移金额

- 输出参数:

字段名	字段	类型	必传	备注

		String	是	交易哈希

3.1.2.2 链账户余额查询

3.1.2.2.1 功能介绍

运营方、平台方或终端用户可以调用方法查询指定账户的余额。

3.1.2.2.2 API 定义

- 方法定义：BigInteger balanceOf(String accAddr);
- 合约方法：balanceOf(address owner) constant returns (uint256 balance);
- 调用者：运营方、平台方或终端用户;
- 核心逻辑：
  1. 检查 accAddr 是有效的合约地址的哈希格式字符串，并且不能为地址;
  2. 检查签名事件是否被注册;
- 输入参数：

字段名	字段	类型	必传	备注
账户地址	accAddr	String	是	查询的账户地址

- 输出参数：

字段名	字段	类型	必传	备注
业务费余额	amount	BigInteger	是	账户所对应的业务费余额

3.1.2.3 DDC 计费规则查询

3.1.2.3.1 功能介绍

运营方、平台方或终端用户可以通过调用该方法查询指定的 DDC 业务合约的方法所对应的调用业务费用。

3.1.2.3.2 API 定义

- 方法定义：BigInteger queryFee(String ddcAddr,String sig);
- 合约方法：queryFee(address ddcAddr,bytes4 sig) returns(uint amount);
- 调用者：运营方、平台方，终端用户；
- 核心逻辑：
  1. 检查 ddcAddr 是有效的合约地址的哈希格式字符串，并且不是空地址；
  2. 检查 sig 是有效的 4 位 Byte 的哈希格式字符串；
  3. 检查签名事件是否被注册；
- 输入参数：

字段名	字段	类型	必传	备注
业务合约地址	ddcAddr	String	是	DDC 业务合约地址
方法 ID	sig	String	是	Hex 格式的合约方法 ID，例如：0x42966c68

- 输出参数：

字段名	字段	类型	必传	备注
业务费	amount	BigInteger	是	查询的 DDC 合约业务费

3.1.3 BSN-DDC-721

3.1.3.1 生成

3.1.3.1.1 功能介绍

平台方或终端用户可以通过调用该方法进行 DDC 的生成。

3.1.3.1.2 API 定义

- 方法定义：String mint(String sender,String to, String ddcURI);
- 合约方法：mint(address to, string memory ddcURI);
- 调用者：平台方、终端用户；
- 核心逻辑：

  1. 检查 sender 为标准备 address 格式；
  2. 检查接收者账户地址信息是否为空；
  3. 检查接收者账户地址格式是否正确；
  4. 检查 DDCURI 信息是否为空；
  5. 检查签名事件是否被注册；

- 输入参数：

字段名	字段	类型	必传	备注
调用者	sender	String	是	调用者地址
接收者账户	to	String	是	
DDC 资源标识符	ddcURI	String	是	

- 输出参数：

字段名	字段	类型	必传	备注

		String	是	交易哈希

3.1.3.2 安全生成

3.1.3.2.1 功能介绍

平台方或终端用户可以通过调用该方法进行 DDC 的安全生成。

3.1.3.2.2 API 定义

- 方法定义：String safeMint(String sender,String to, String ddcURI, byte[] data);
- 合约方法：safeMint(address to, string memory ddcURI,bytes memory data);
- 调用者：平台方、终端用户；
- 核心逻辑：
  1. 检查 sender 为标准备 address 格式；
  2. 检查接收者账户地址信息是否为空；
  3. 检查接收者账户地址格式是否正确；
  4. 检查 DDCURI 信息是否为空；
  5. 检查签名事件是否被注册；
- 输入参数：

字段名	字段	类型	必传	备注
调用者	sender	String	是	调用者地址
接收者账户	to	String	是	
DDC 资源标识符	ddcURI	String	是	

附加数据	data	byte[]	否	

- 输出参数:

字段名	字段	类型	必传	备注
		String	是	交易哈希

3.1.3.3 DDC 授权

3.1.3.3.1 功能介绍

DDC 拥有者可以通过调用该方法进行 DDC 的授权，发起者需要是 DDC 的拥有者。

3.1.3.3.2 API 定义

- 方法定义: String approve(String sender,String to,BigInteger ddcId);
- 合约方法: approve(address to,uint256 ddcId);
- 调用者: DDC 拥有者;
- 核心逻辑:

1. 检查 sender 为标准备 address 格式;
2. 检查授权者账户地址信息是否为空;
3. 检查授权者账户地址格式是否正确;
4. 检查 ddcId 是否大于 0;
5. 检查签名事件是否被注册;

- 输入参数:

字段名	字段	类型	必传	备注
调用者	sender	String	是	调用者地址
授权者账户	to	String	是	

DDC 唯一标识	ddcId	BigInteger	是	

- 输出参数:

字段名	字段	类型	必传	备注
		String	是	交易哈希

3.1.3.4 DDC 授权查询

3.1.3.4.1 功能介绍

运营方、平台方或终端用户可以通过调用该方法进行 DDC 的授权查询。

3.1.3.4.2 API 定义

- 方法定义: String getApproved(BigInteger ddcId);
- 合约方法: getApproved(uint256 ddcId);
- 调用者: 运营方、平台方或终端用户;
- 核心逻辑:

1. 检查 ddcId 的值是否大于 0;
2. 检查签名事件是否被注册;

- 输入参数:

字段名	字段	类型	必传	备注
DDC 唯一标识	ddcId	BigInteger	是	

- 输出参数:

字段名	字段	类型	必传	备注
授权的账户	operator	String	是	

3.1.3.5 账户授权

3.1.3.5.1 功能介绍

DDC 拥有者可以通过调用该方法进行账户授权，发起者需要是 DDC 的拥有者。

3.1.3.5.2 API 定义

- 方法定义：String setApprovalForAll(String sender,String operator,bool approved);
- 合约方法：setApprovalForAll(address operator,bool approved);
- 调用者：DDC 拥有者;
- 核心逻辑：
  1. 检查 sender 为标准备 address 格式;
  2. 检查授权者账户地址信息是否为空;
  3. 检查授权者账户地址格式是否正确;
  4. 检查签名事件是否被注册;
- 输入参数：

字段名	字段	类型	必传	备注
调用者	sender	String	是	调用者地址
授权者账户	operator	String	是	
授权标识	approved	Boolean	是	

- 输出参数：

字段名	字段	类型	必传	备注
		String	是	交易哈希

3.1.3.6 账户授权查询

3.1.3.6.1 功能介绍

运营方、平台方或终端用户可以通过调用该方法进行账户授权查询。

3.1.3.6.2 API 定义

- 方法定义：Boolean isApprovedForAll(String owner,String operator);
- 合约方法：isApprovedForAll(address owner,address operator) returns (bool);
- 调用者：运营方、平台方或终端用户;
- 核心逻辑：
  1. 检查拥有者账户地址信息是否为空;
  2. 检查拥有者账户地址格式是否正确;
  3. 检查授权者账户地址信息是否为空;
  4. 检查授权者账户地址格式是否正确;
  5. 检查签名事件是否被注册;
- 输入参数：

字段名	字段	类型	必传	备注
拥有者账户	owner	String	是	
授权者账户	operator	String	是	

- 输出参数：

字段名	字段	类型	必传	备注
授权标识	approved	Boolean	是	

3.1.3.7 安全转移

3.1.3.7.1 功能介绍

DDC 的拥有者或授权者可以通过调用该方法进行 DDC 的转移。

3.1.3.7.2 API 定义

- 方法定义：String safeTransferFrom(String sender,String from,String to,BigInteger ddcId, byte[] data);
- 合约方法：safeTransferFrom(address from,address to,uint256 ddcId, byte[] data);
- 调用者：DDC 拥有者、DDC 授权者;
- 核心逻辑：
  1. 检查 sender 为标准备 address 格式
  2. 检查拥有者账户地址信息是否为空;
  3. 检查拥有者账户地址格式是否正确;
  4. 检查接收者账户地址信息是否为空;
  5. 检查接收者账户地址格式是否正确;
  6. 检查 ddcId 的数值是否大于 0;
  7. 检查签名事件是否被注册;
- 输入参数：

字段名	字段	类型	必传	备注
调用者	sender	String	是	调用者地址
拥有者账户	from	String	是	
接收者账户	to	String	是	
DDC 唯一标识	ddcId	BigInteger	是	

附加数据	data	Byte[]	否	

- 输出参数:

字段名	字段	类型	必传	备注
		String	是	交易哈希

3.1.3.8 转移

3.1.3.8.1 功能介绍

DDC 拥有者或授权者可以通过调用该方法进行 DDC 的转移。

3.1.3.8.2 API 定义

- 方法定义: String transferFrom(String sender,String from,String to,BigInteger ddcId);
- 合约方法: transferFrom(address from,address to,uint256 ddcId);
- 调用者: DDC 拥有者、DDC 授权者;
- 核心逻辑:

1. 检查 sender 为标准备 address 格式;
2. 检查拥有者账户地址信息是否为空;
3. 检查拥有者账户地址格式是否正确;
4. 检查接收者账户地址信息是否为空;
5. 检查接收者账户地址格式是否正确;
6. 检查 ddcId 的数值是否大于 0;
7. 检查签名事件是否被注册;

- 输入参数:

字段名	字段	类型	必传	备注

调用者	sender	String	是	调用者地址
拥有者账户	from	String	是	
接收者账户	to	String	是	
DDC 唯一标识	ddcId	BigInteger	是	

- 输出参数：

字段名	字段	类型	必传	备注
		String	是	交易哈希

3.1.3.9 销毁

3.1.3.9.1 功能介绍

DDC 拥有者或 DDC 授权者可以通过调用该方法进行 DDC 的销毁。

3.1.3.9.2 API 定义

- 方法定义：String burn(String sender,BigInteger ddcId);
- 合约方法：burn(uint256 ddcId);
- 调用者：DDC 拥有者、DDC 授权者；
- 核心逻辑：

1. 检查 sender 为标准备 address 格式；
2. 检查 ddcId 的数值是否大于 0；
3. 检查签名事件是否被注册；

- 输入参数：

字段名	字段	类型	必传	备注
调用者	sender	String	是	调用者地址
DDC 唯一标识	ddcId	BigInteger	是	

- 输出参数:

字段名	字段	类型	必传	备注
		String	是	交易哈希

3.1.3.10 查询数量

3.1.3.10.1 功能介绍

运营方、平台方以及终端用户可以通过调用该方法进行查询当前账户拥有的 DDC 的数量。

3.1.3.10.2 API 定义

- 方法定义：BigInteger balanceOf(String owner);
- 合约方法：balanceOf(address owner) returns (uint256);
- 调用者：运营方、平台方以及终端用户;
- 核心逻辑:

1. 检查拥有者账户地址信息是否为空;
2. 检查拥有者账户地址格式是否正确;
3. 检查签名事件是否被注册;

- 输入参数:

字段名	字段	类型	必传	备注
拥有者账户	owner	String	是	

- 输出参数:

字段名	字段	类型	必传	备注
DDC 的数量	balance	BigInteger	是	

3.1.3.11 查询拥有者

3.1.3.11.1 功能介绍

运营方、平台方以及终端用户可以通过调用该方法查询当前 DDC 的拥有者。

3.1.3.11.2 API 定义

- 方法定义：String ownerOf(BigInteger ddcId);
- 合约方法：ownerOf(uint256 ddcId) returns (address);
- 调用者：运营方、平台方以及终端用户;
- 核心逻辑：
  1. 检查 ddcId 的数值是否大于 0;
  2. 检查签名事件是否被注册;
- 输入参数：

字段名	字段	类型	必传	备注
DDC 唯一标识	ddcId	BigInteger	是	

- 输出参数：

字段名	字段	类型	必传	备注
拥有者账户	owner	String	是	

3.1.3.12 获取名称

3.1.3.12.1 功能介绍

运营方、平台方以及终端用户可以通过调用该方法查询当前 DDC 的名称。

3.1.3.12.2 API 定义

- 方法定义：String Name();
- 合约方法：name() returns (string memory);
- 调用者：运营方、平台方以及终端用户;
- 核心逻辑：

  1. 检查签名事件是否被注册;

- 输入参数：

字段名	字段	类型	必传	备注

- 输出参数：

字段名	字段	类型	必传	备注
		String	是	DDC 运营方名称

3.1.3.13 获取符号

3.1.3.13.1 功能介绍

运营方、平台方以及终端用户可以通过调用该方法查询当前 DDC 的符号标识。

3.1.3.13.2 API 定义

- 方法定义：String symbol();
- 合约方法：symbol() returns (string memory);
- 调用者：运营方、平台方以及终端用户;

- 核心逻辑:

  1. 检查签名事件是否被注册;

- 输入参数:

字段名	字段	类型	必传	备注

- 输出参数:

字段名	字段	类型	必传	备注
		String	是	DDC 运营方符号

3.1.3.14 获取 DDCURI

3.1.3.14.1 功能介绍

运营方、平台方以及终端用户可以通过调用该方法查询当前 DDC 的资源标识符。

3.1.3.14.2 API 定义

- 方法定义: String ddcURI(BigInteger ddcId);
- 合约方法: ddcURI(uint256 ddcId) returns (string memory);
- 调用者: 运营方、平台方以及终端用户;
- 核心逻辑:

  1. 检查 ddcId 的数值是否大于 0;
  2. 检查签名事件是否被注册;

- 输入参数:

字段名	字段	类型	必传	备注

DDC 唯一标识	ddcId	BigInteger	是	

- 输出参数:

字段名	字段	类型	必传	备注
DDC 资源标识符	ddcURI	String	是	

3.1.3.15 URI 设置

3.1.3.15.1 功能介绍

DDC 拥有者或 DDC 授权者通过调用该方法对 DDC 的资源标识符进行设置。

3.1.3.15.2 API 定义

- 方法定义: String setURI(String sender,BigInteger ddcId,String ddcURI);
- 合约方法: setURI(uint256 ddcId,string memory ddcURI);
- 调用者: DDC 拥有者、DDC 授权者;
- 核心逻辑:

1. 检查 sender 为标准备 address 格式;
2. 检查 ddcId 的数值是否大于 0;
3. 检查 ddcURI 是否为空字符串;
4. 检查签名事件是否被注册;

- 输入参数:

字段名	字段	类型	必传	备注
调用者	sender	String	是	调用者地址
DDC 唯一标识	ddcId	BigInteger	是	

DDC 资源标识符	ddcURI	String	是	

➢ 输出参数:

字段名	字段	类型	必传	备注
		String	是	交易哈希

3.1.4 BSN-DDC-1155

3.1.4.1 安全生成

3.1.4.1.1 功能介绍

平台方或终端用户可以通过调用该方法进行 DDC 的安全生成。

3.1.4.1.2 API 定义

➢ 方法定义: String safeMint(String sender,String to,BigInteger amount,String ddcURI,byte[] data);

➢ 合约方法: safeMint(address to,uint256 amount, string memory ddcURI, bytes memory data);

➢ 调用者: 平台方、终端用户;

➢ 核心逻辑:

1. 检查 sender 为标准备 address 格式;
2. 检查接收者账户地址信息是否为空;
3. 检查接收者账户地址格式是否正确;
4. 检查需要生成的 DDC 数量是否大于 0;
5. 检查 DDCURI 信息是否为空;
6. 检查签名事件是否被注册;

- 输入参数:

字段名	字段	类型	必传	备注
调用者	sender	String	是	调用者地址
接收者账户	to	String	是	
DDC 数量	amount	BigInteger	是	
DDCURI	ddcURI	String	是	
附加数据	data	byte[]	否	

- 输出参数:

字段名	字段	类型	必传	备注
		String	是	交易哈希

3.1.4.2 批量安全生成

3.1.4.2.1 功能介绍

平台方或终端用户可以通过调用该方法进行 DDC 的批量安全生成。

3.1.4.2.2 API 定义

- 方法定义: String safeMintBatch(String sender,String to,Multimap<BigInteger,String> ddcInfo,byte[] data);
- 合约方法: safeMintBatch(address to,uint256[] amounts, string[] ddcURIs, bytes memory data);
- 调用者: 平台方、终端用户;
- 核心逻辑:

1. 检查 sender 为标准备 address 格式;

2. 检查接收者账户地址信息是否为空;
3. 检查接收者账户地址格式是否正确;
4. 检查生成的 DDC 数量集合大小是否大于 0;
5. 检查生成的 DDC 数量集合中每个 DDC 数量是否大于 0;
6. 检查生成的 DDCURI 集合大小是否大于 0;
7. 检查生成的 DDCURI 集合中每个 DDCURI 是否为空;
8. 检查生成的 DDC 数量集合与 DDCURI 集合的大小是否相等;
9. 检查签名事件是否被注册;

➢ 输入参数:

字段名	字段	类型	必传	备注
调用者	sender	String	是	调用者地址
接收者账户	to	String	是	
DDC 信息合计	ddcInfo	Multimap<BigInteger,String>	是	
附加数据	data	byte[]	否	

➢ 输出参数:

字段名	字段	类型	必传	备注
		String	是	交易哈希

3.1.4.3 账户授权

3.1.4.3.1 功能介绍

DDC 拥有者可以通过调用该方法进行账户授权，发起者需要是 DDC 的拥有者。

3.1.4.3.2 API 定义

- 方法定义：String setApprovalForAll(String sender,String operator, Boolean approved);
- 合约方法：setApprovalForAll(address operator, bool approved);
- 调用者：DDC 拥有者;
- 核心逻辑：
  1. 检查 sender 为标准备 address 格式;
  2. 检查授权者账户地址信息是否为空;
  3. 检查授权者账户地址格式是否正确;
  4. 检查签名事件是否被注册;
- 输入参数：

字段名	字段	类型	必传	备注
调用者	sender	String	是	调用者地址
授权者账户	operator	String	是	
授权标识	approved	Boolean	是	

- 输出参数：

字段名	字段	类型	必传	备注
		String	是	交易哈希

3.1.4.4 账户授权查询

3.1.4.4.1 功能介绍

运营方、平台方或终端用户可以通过调用该方法进行账户授权查询。

3.1.4.4.2 API 定义

- 方法定义：Boolean isApprovedForAll(String owner,String operator);
- 合约方法：isApprovedForAll(address owner,address operator) returns (bool);
- 调用者：运营方、平台方或终端用户;
- 核心逻辑：
  1. 检查拥有者账户地址信息是否为空;
  2. 检查拥有者账户地址格式是否正确;
  3. 检查授权者账户地址信息是否为空;
  4. 检查授权者账户地址格式是否正确;
  5. 检查签名事件是否被注册;
- 输入参数：

字段名	字段	类型	必传	备注
拥有者账户	owner	String	是	
授权者账户	operator	Boolean	是	

- 输出参数：

字段名	字段	类型	必传	备注
		Boolean	是	

3.1.4.5 安全转移

3.1.4.5.1 功能介绍

DDC 拥有者或 DDC 授权者可以通过调用该方法进行 DDC 的转移。

3.1.4.5.2 API 定义

- 方法定义：String safeTransferFrom(String sender,String from,String to,BigInteger ddcId,BigInteger amount,byte[] data);
- 合约方法：safeTransferFrom(address from,address to,uint256 ddcId,uint256 amount,bytes memory data);
- 调用者：DDC 拥有者、DDC 授权者;
- 核心逻辑：

1. 检查 sender 为标准备 address 格式;
2. 检查拥有者账户地址信息是否为空;
3. 检查拥有者账户地址格式是否正确;
4. 检查接收者账户地址信息是否为空;
5. 检查接收者账户地址格式是否正确;
6. 检查 DDCID 数值是否大于 0;
7. 检查 DDC 转移所对应的数量是否大于 0;
8. 检查签名事件是否被注册;

- 输入参数：

字段名	字段	类型	必传	备注
调用者	sender	String	是	调用者地址
拥有者账户	from	String	是	
接收者账户	to	String	是	
DDCID	ddcId	BigInteger	是	
数量	amount	BigInteger	是	DDCID 所对应的数量
附加数据	data	byte[]	否	

- 输出参数：

字段名	字段	类型	必传	备注
		String	是	交易哈希

3.1.4.6 批量安全转移

3.1.4.6.1 功能介绍

DDC 拥有者或 DDC 授权者可以通过调用该方法进行 DDC 的批量转移。

3.1.4.6.2 API 定义

- 方法定义：String safeBatchTransferFrom(String sender,String from,String to,Map<BigInteger,BigInteger> ddcs, byte[] data);
- 合约方法：safeBatchTransferFrom(address from, address to,uint256[] ddclds,uint256[] amounts,bytes memory data);
- 调用者：DDC 拥有者、DDC 授权者;
- 核心逻辑:
    1. 检查 sender 为标准备 address 格式;
    2. 检查拥有者账户地址信息是否为空;
    3. 检查拥有者账户地址格式是否正确;
    4. 检查接收者账户地址信息是否为空;
    5. 检查接收者账户地址格式是否正确;
    6. 检查转移的 ddcs 集合大小是否大于 0;
    7. 检查转移的 ddcs 集合中每个 DDCID 是否大于 0;
    8. 检查转移的 ddcs 集合中每个 DDC 数量是否大于 0;
    9. 检查签名事件是否被注册;
- 输入参数:

字段名	字段	类型	必传	备注
调用者	sender	String	是	调用者地址
拥有者账户	from	String	是	
接收者账户	to	String	是	
拥有者 DDCID 集合	ddcs	Map<BigInteger,BigInteger>	是	
附加数据	data	byte[]	否	

- 输出参数：

字段名	字段	类型	必传	备注
		String	是	交易哈希

3.1.4.7 销毁

3.1.4.7.1 功能介绍

DDC 拥有者可以通过调用该方法进行 DDC 的销毁。

3.1.4.7.2 API 定义

- 方法定义：String burn(String sender,String owner,BigInteger ddcId);
- 合约方法：burn(address owner,uint256 ddcId);
- 调用者：DDC 拥有者；
- 核心逻辑：

1. 检查 sender 为标准备 address 格式；
2. 检查拥有者账户地址信息是否为空；
3. 检查拥有者账户地址格式是否正确；
4. 检查需要销毁的 DDCID 集合长度是否大于 0；

5. 检查签名事件是否被注册;

- 输入参数:

字段名	字段	类型	必传	备注
调用者	sender	String	是	调用者地址
拥有者账户	owner	String	是	
DDCID	ddcId	BigInteger	是	

- 输出参数:

字段名	字段	类型	必传	备注
		String	是	交易哈希

3.1.4.8 批量销毁

3.1.4.8.1 功能介绍

DDC 拥用者可以通过调用该方法进行 DDC 的批量销毁。

3.1.4.8.2 API 定义

- 方法定义: String burnBatch(String sender,String owner,List<BigInteger> ddcIds);
- 合约方法: burnBatch(address owner,uint256[] ddcIds);
- 调用者: DDC 拥用者;
- 核心逻辑:

1. 检查 sender 为标准备 address 格式;
2. 检查拥有者账户地址信息是否为空;
3. 检查拥有者账户地址格式是否正确;
4. 检查需要销毁的 DDCID 集合大小是否大于 0;

5. 检查需要销毁的 DDCID 集合中每个 DDCID 数值是否大于 0;

6. 检查签名事件是否被注册;

- 输入参数:

字段名	字段	类型	必传	备注
调用者	sender	String	是	调用者地址
拥有者账户	owner	String	是	
DDCID 集合	ddclds	List<BigInteger>	是	

- 输出参数:

字段名	字段	类型	必传	备注
		String	是	交易哈希

3.1.4.9 查询数量

3.1.4.9.1 功能介绍

运营方、平台方以及终端用户可以通过调用该方法进行查询当前账户拥有的 DDC 的数量。

3.1.4.9.2 API 定义

- 方法定义: BigInteger balanceOf(String owner,BigInteger ddcId);
- 合约方法: balanceOf(address owner, uint256 ddcId) returns (uint256);
- 调用者: 运营方、平台方以及终端用户;
- 核心逻辑:

1. 检查拥有者账户地址信息是否为空;
2. 检查拥有者账户地址格式是否正确;
3. 检查 DDCID 集合长度是否大于 0;

4. 检查签名事件是否被注册;

- 输入参数:

字段名	字段	类型	必传	备注
拥有者账户	owner	String	是	
DDCID	ddcId	BigInteger	是	

- 输出参数:

字段名	字段	类型	必传	备注
数量		BigInteger	是	拥有者账户所对应的 DDCID 所拥用的数量

3.1.4.10 批量查询数量

3.1.4.10.1 功能介绍

运营方、平台方以及终端用户可以通过调用该方法进行批量查询账户拥有的 DDC 的数量。

3.1.4.10.2 API 定义

- 方法定义: List<BigInteger> balanceOfBatch(Multimap<String,BigInteger> ddcs);
- 合约方法: balanceOfBatch(address[] memory owners,uint256[] memory ddcIds) returns (uint256[] memory);
- 调用者: 运营方、平台方以及终端用户;
- 核心逻辑:

1. 检查 ddcs 集合大小是否大于 0;

2. 检查 ddcs 集合中拥有者账户地址信息是否为空;
3. 检查 ddcs 集合中拥有者账户地址格式是否正确;
4. 检查 ddcs 集合中每个 DDCID 数值是否大于 0;
5. 检查签名事件是否被注册;

- 输入参数:

字段名	字段	类型	必传	备注
拥有者 DDCID 集合	ddcs	Multimap<String,BigInteger>	是	

- 输出参数:

字段名	字段	类型	必传	备注
数量集合		List<BigInteger>	是	拥有者账户所对应的每个 DDCID 所拥用的数量

3.1.4.11 获取 DDCURI

3.1.4.11.1 功能介绍

运营方、平台方以及终端用户可以通过调用该方法进行查询当前 DDC 的资源标识符。

3.1.4.11.2 API 定义

- 方法定义: String ddcURI(BigInteger ddcId);
- 合约方法: ddcURI(uint256 ddcId) returns (string memory);
- 调用者: 运营方、平台方以及终端用户;
- 核心逻辑:

1. 检查 DDCID 数值是否大于 0;
2. 检查签名事件是否被注册;

- 输入参数:

字段名	字段	类型	必传	备注
DDCID	ddcId	BigInteger	是	

- 输出参数:

字段名	字段	类型	必传	备注
DDCURI		String	是	

3.1.4.12 URI 设置

3.1.4.12.1 功能介绍

DDC 拥有者或 DDC 授权者通过调用该方法对 DDC 的资源标识符进行设置。

3.1.4.12.2 API 定义

- 方法定义: String setURI(String sender,String owner,BigInteger ddcId,String ddcURI);
- 合约方法: setURI(address owner,uint256 ddcId,string memory ddcURI);
- 调用者: DDC 拥有者、DDC 授权者;
- 核心逻辑:

1. 检查 sender 为标准备 address 格式;
2. 检查 owner 地址格式是否正确;
3. 检查 ddcId 数值是否大于 0;
4. 检查 ddcURI 是否为空字符串;
5. 检查签名事件是否被注册;

- 输入参数:

字段名	字段	类型	必传	备注
调用者	sender	String	是	调用者地址
DDC 拥有者	owner	String	是	
DDC 唯一标识	ddcId	BigInteger	是	
DDC 资源标识符	ddcURI	String	是	

- 输出参数:

字段名	字段	类型	必传	备注
		String	是	交易哈希

3.1.5 BSN-DDC-交易查询

3.1.5.1 查询交易信息

3.1.5.1.1 功能介绍

运营方或平台方根据交易哈希对交易信息进行查询。

3.1.5.1.2 API 定义

- 方法定义: String getTransByHash(String txHash)
- 调用者: 平台方、运营方;
- 核心逻辑:

  1. 根据交易哈希查询的交易信息（不同框架自定义输出参数）
  2. 检查签名事件是否被注册;

- 输入参数:

字段名	字段	类型	必传	备注

交易哈希	txHash	String	是	

- 输出参数:

字段名	字段	类型	必传	备注
交易信息	txInfo	String	是	

3.1.5.2 查询交易回执

3.1.5.2.1 功能介绍

运营方或平台方根据交易哈希对交易回执信息进行查询。

3.1.5.2.2 API 定义

- 方法定义: String getTransReceipt(String txHash)
- 调用者: 平台方、运营方;
- 核心逻辑:

1. 根据交易 hash 查询交易回执（不同框架自定义输出参数）
2. 检查签名事件是否被注册;

- 输入参数:

字段名	字段	类型	必传	备注
交易哈希	txHash	String	是	

- 输出参数:

字段名	字段	类型	必传	备注
交易回执	txReceipt	String	是	

3.1.5.3 查询交易状态

3.1.5.3.1 功能介绍

运营方或平台方根据交易哈希查询交易状态是否成功。

3.1.5.3.2 API 定义

- 方法定义：Boolean getTransByStatus(String txHash)
- 调用者：平台方、运营方；
- 核心逻辑：

  1. 根据交易哈希查询的交易是否成功；
  2. 检查签名事件是否被注册；

- 输入参数：

字段名	字段	类型	必传	备注
交易哈希	txHash	String	是	

- 输出参数：

字段名	字段	类型	必传	备注
交易是否成功	txStatus	Boolean	是	

3.1.6 BSN-DDC-区块查询

3.1.6.1 获取区块信息

3.1.6.1.1 功能介绍

运营方或平台方根据区块高度对区块信息进行查询，并解析区块数据返回给运营方或平台方。

3.1.6.1.2 API 定义

- 方法定义：String getBlockByNumber(BigInteger blockNumber)
- 调用者：运营方、平台方；
- 核心逻辑：

  1. 根据区块高度查询区块信息（不同框架自定义输出参数）；
  2. 检查签名事件是否被注册；

- 输入参数：

字段名	字段	类型	必传	备注
区块高度	blockNumber	BigInteger	是	

- 输出参数：

字段名	字段	类型	必传	备注
区块信息	blockInfo	String	是	

3.1.7 BSN-DDC-签名事件

3.1.7.1 功能介绍

此事件是通用事件，所有的上链待签名交易报文需调用此事件进行签名，业务调用方需要注册此签名事件，并在实现的签名事件中实现签名逻辑，并将最终签名后的结果返回给 DDC-SDK。

3.1.7.2 事件定义

- 输入参数：签名事件类
- 输出参数：签名结果
- String signEvent(SignEvent event);

3.1.7.3 数据结构

- SignEvent

字段名	字段	类型	必传	备注
签名者	sender	String	是	
待签名交易	unSignTrans	Object	是	

3.1.8 BSN-DDC-数据解析

3.1.8.1 权限数据

3.1.8.1.1 添加账户

3.1.8.1.1.1 功能说明

用于对 BSN-DDC-权限合约进行添加账户所产生的交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.1.1.2 合约事件

- AddAccount (address indexed caller,address indexed account)

3.1.8.1.1.3 数据结构

字段名	字段	类型	必传	备注
签名者	sender	String	是	
链账户地址	account	String	是	添加的链账户地址

3.1.8.1.2 更新账户状态

3.1.8.1.2.1 功能说明

用于对 BSN-DDC-权限合约进行更新账户状态所产生的交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.1.2.2 合约事件

- UpdateAccountState(address indexed account,IAuthorityData.State platformState,IAuthorityData.State operatorState)

3.1.8.1.2.3 数据结构

字段名	字段	类型	必传	备注
链账户地址	account	String	是	添加的链账户地址
平台管理账户状态	platformState	enum	是	DDC 账户对应的当前账户状态（仅平台方可操作该状态）。值包含: 1.Frozen（冻结状态，无法进行 DDC 相关操作） 2.Active （活跃状态，可进行 DDC 相关操作）
运营管理账户状态	operatorState	enum	是	DDC 账户对应的当前账户状态（仅运营方可操作该状态）。值包含: 1.Frozen（冻结状态，无法进行 DDC 相关操作） 2.Active （活跃状态，可进行 DDC 相

				关操作）

3.1.8.1.3 跨平台授权

3.1.8.1.3.1 功能说明

用于对 BSN-DDC-权限合约进行跨平台授权所产生的交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.1.3.2 合约事件

- CrossPlatformApproval(address indexed from,address indexed to, bool approved)

3.1.8.1.3.3 数据结构

字段名	字段	类型	必传	备注
授权账户	from	String	是	
接收账户	to	String	是	
授权标识	approved	Boolean	是	

3.1.8.2 充值数据

3.1.8.2.1 充值

3.1.8.2.1.1 功能说明

用于对 BSN-DDC-计费合约进行充值所产生的交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.2.1.2 合约事件

- Recharge(address indexed from,address indexed to,uint256 value)

3.1.8.2.1.3 数据结构

字段名	字段	类型	必传	备注
原链账户地址	from	String	是	业务费转出方链账户地址
目标链账户地址	to	String	是	业务费转入方链账户地址
业务费	amount	BigInteger	是	充值的业务费金额

3.1.8.2.2 DDC 业务费扣除

3.1.8.2.2.1 功能说明

用于对 BSN-DDC-计费合约进行 DDC 业务费扣除所产生的交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.2.2.2 合约事件

- Pay(address indexed from,address indexed ddcAddr,bytes4 sig,uint32 amount,uint256 ddcId)

3.1.8.2.2.3 数据结构

字段名	字段	类型	必传	备注
链账户地址	from	String	是	扣除业务费方链账户地址
业务合约	ddcAddr	String	是	业务合约地址

方法签名	sig	String	是	业务合约所对应的方法签名
业务费	amount	BigInteger	是	充值的业务费金额
DDC 唯一标识	ddcId	BigInteger	是	

3.1.8.3 BSN-DDC-721 数据

3.1.8.3.1 生成/安全生成

3.1.8.3.1.1 功能说明

用于对 BSN-DDC-721 业务合约进行 DDC 生成或安全生成所产生的交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.3.1.2 合约事件

- Transfer(address(0),to,ddcId)

3.1.8.3.1.3 数据结构

字段名	字段	类型	必传	备注
接收账户地址	to	String	是	
DDCID	ddcId	BigInteger	是	

3.1.8.3.2 转移/安全转移

3.1.8.3.2.1 功能说明

用于对 BSN-DDC-721 业务合约进行 DDC 转移/安全转移所产生的交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.3.2.2 合约事件

- Transfer(from, to, ddcId)

3.1.8.3.2.3 数据结构

字段名	字段	类型	必传	备注
拥有账户地址	from	String	是	
接收账户地址	to	String	是	
DDCID	ddcId	BigInteger	是	

3.1.8.3.3 冻结

3.1.8.3.3.1 功能说明

用于对 BSN-DDC-721 业务合约进行 DDC 解冻所产生的交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.3.3.2 合约事件

- EnterBlacklist(sender,ddcId)

3.1.8.3.3.3 数据结构

字段名	字段	类型	必传	备注
签名者	sender	String	是	签名者账户所对应的账户地址
DDCID	ddcId	BigInteger	是	

3.1.8.3.4 解冻

3.1.8.3.4.1 功能说明

用于对 BSN-DDC-721 业务合约进行 DDC 解冻所产生的交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.3.4.2 合约事件

- ExitBlacklist(sender,ddcId)

3.1.8.3.4.3 数据结构

字段名	字段	类型	必传	备注
签名者	sender	String	是	签名者账户所对应的账户地址
DDCID	ddcId	BigInteger	是	

3.1.8.3.5 销毁

3.1.8.3.5.1 功能说明

用于对 BSN-DDC-721 业务合约进行 DDC 销毁所产生的交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.3.5.2 合约事件

- Transfer(from,address(0),ddcId)

3.1.8.3.5.3 数据结构

字段名	字段	类型	必传	备注
拥有者	from	String	是	DDC 拥有者所对应的账户地址
DDCID	ddcId	BigInteger	是	

3.1.8.3.6 URI 设置

3.1.8.3.6.1 功能说明

用于对 BSN-DDC-721 业务合约进行 DDC 资源标识符设置所产生的交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.3.6.2 合约事件

- SetURI(uint256 indexed ddcId,string ddcURI)

3.1.8.3.6.3 数据结构

字段名	字段	类型	必传	备注
DDC 唯一标识	ddcId	BigInteger	是	
DDC 资源标识符	ddcURI	String	是	

3.1.8.4 BSN-DDC-1155 数据

3.1.8.4.1 安全生成

3.1.8.4.1.1 功能说明

用于对 BSN-DDC-1155 业务合约进行 DDC 安全生成所产生的交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.4.1.2 合约事件

- TransferSingle(operator,address(0),to,ddcId,amount)

3.1.8.4.1.3 数据结构

字段名	字段	类型	必传	备注
签名者	operator	String	是	签名者账户所对

				应的账户地址
接收账户地址	to	String	是	
DDCID	ddcId	BigInteger	是	
数量	amount	BigInteger	是	

3.1.8.4.2 批量安全生成

3.1.8.4.2.1 功能说明

用于对 BSN-DDC-1155 业务合约进行 DDC 批量安全生成所产生的交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.4.2.2 合约事件

- TransferBatch(operator,address(0),to,ddcIds,amounts)

3.1.8.4.2.3 数据结构

字段名	字段	类型	必传	备注
签名者	operator	String	是	签名者账户所对应的账户地址
接收账户地址	to	String	是	
DDC 集合	ddcs	Map<BigInteger, BigInteger>	是	Key：ddcId Value:amount

3.1.8.4.3 安全转移

3.1.8.4.3.1 功能说明

用于对 BSN-DDC-1155 业务合约进行 DDC 安全转移所产生的

交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.4.3.2 合约事件

- TransferSingle(operator,from,to,ddcId,amount)

3.1.8.4.3.3 数据结构

字段名	字段	类型	必传	备注
签名者	operator	String	是	签名者账户所对应的账户地址
拥有账户地址	from	String	是	
接收账户地址	to	String	是	
DDCID	ddcId	BigInteger	是	
数量	amount	BigInteger	是	

3.1.8.4.4 批量安全转移

3.1.8.4.4.1 功能说明

用于对 BSN-DDC-1155 业务合约进行 DDC 批量安全转移所产生的交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.4.4.2 合约事件

- TransferBatch(operator,from,to,ddcIds,amounts)

3.1.8.4.4.3 数据结构

字段名	字段	类型	必传	备注
签名者	operator	String	是	签名者账户所对应的账户地址

拥有账户地址	from	String	是	
接收账户地址	to	String	是	
DDC 集合	ddcs	Map<BigInteger, BigInteger>	是	Key：ddcId Value:amount

3.1.8.4.5 冻结

3.1.8.4.5.1 功能说明

用于对 BSN-DDC-1155 业务合约进行 DDC 冻结所产生的交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.4.5.2 合约事件

- EnterBlacklist(sender,ddcId)

3.1.8.4.5.3 数据结构

字段名	字段	类型	必传	备注
签名者	sender	String	是	签名者账户所对应的账户地址
DDCID	ddcId	BigInteger	是	

3.1.8.4.6 解冻

3.1.8.4.6.1 功能说明

用于对 BSN-DDC-1155 业务合约进行 DDC 解冻所产生的交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.4.6.2 合约事件

- ExitBlacklist(sender,ddcId)

3.1.8.4.6.3 数据结构

字段名	字段	类型	必传	备注
签名者	sender	String	是	签名者账户所对应的账户地址
DDCID	ddcId	BigInteger	是	

3.1.8.4.7 销毁

3.1.8.4.7.1 功能说明

用于对 BSN-DDC-1155 业务合约进行 DDC 销毁所产生的交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.4.7.2 合约事件

- TransferSingle(operator,from,address(0),ddcId,amount)

3.1.8.4.7.3 数据结构

字段名	字段	类型	必传	备注
签名者	operator	String	是	签名者账户所对应的账户地址
拥有账户地址	from	String	是	
DDCID	ddcId	BigInteger	是	
数量	amount	BigInteger	是	

3.1.8.4.8 批量销毁

3.1.8.4.8.1 功能说明

用于对 BSN-DDC-1155 业务合约进行 DDC 批量销毁所产生的交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.4.8.2 合约事件

- TransferBatch(operator,from,address(0),ddcIds,amounts)

3.1.8.4.8.3 数据结构

字段名	字段	类型	必传	备注
签名者	operator	String	是	签名者账户所对应的账户地址
拥有账户地址	from	String	是	
DDC 集合	ddcs	Map<BigInteger, BigInteger>	是	Key：ddcId Value:amount

3.1.8.4.9 URI 变更

3.1.8.4.9.1 功能说明

用于对 BSN-DDC-1155 业务合约进行 DDC 资源标识符设置所产生的交易回执中的事件进行解析，并组装成所对应的数据结构。

3.1.8.4.9.2 合约事件

- SetURI(address indexed owner,uint256 indexed ddcId,string ddcURI)

3.1.8.4.9.3 数据结构

字段名	字段	类型	必传	备注
拥有者	owner	String	是	
DDC 唯一标识	ddcId	BigInteger	是	
DDC 资源标识符	ddcURI	String	是	

4. 附录

4.1 区块信息示例

4.1.1 泰安链


```
{
        "dbHash":
"0x0000000000000000000000000000000000000000000000000000000000000000
00",
        "extraData": [],
        "gasLimit": "0x0",
        "gasUsed": "0x0",
        "hash":
"0xfa639d1454362a8cdfcab1ca1948a5defaf7048b28f67e80780ab1e24e8f8c59",
        "logsBloom":
"0x0000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000
00",
        "number": "0x1",
        "parentHash":
"0x249f59e00beac8424a7821c4750fdd70c128f4ce795afbab53f345e9fce95d1a",
        "receiptsRoot":
"0x69a04fa6073e4fc0947bac7ee6990e788d1e2c5ec0fe6c2436d0892e7f3c09d2",
        "sealer": "0x0",
        "sealerList": [

"4ca3a91a4937355dba6a2e5fe76141479a1fc44e9caa86750092dab64e0b8382f6
b8476749c2d2de414350a54491620d38813d2a1442f524e36e3d9946109c4d"
        ],
        "signatureList": [
          {
            "index": "0x0",
            "signature":
"0x4602135870d9a4846e2536d4a48e831825a5d95768dd0d4f08544a0bd4c2af
```

41242dec1751a05c07d7572027f8d6ac1625c48145beb004e2dce8b7ce9e2bb73d00"
        }
      ],
      "stateRoot": "0x0000000000000000000000000000000000000000000000000000000000000000",
      "timestamp": "0x175ac38cf10",
      "transactions": [
        {
          "blockHash": "0xfa639d1454362a8cdfcab1ca1948a5defaf7048b28f67e80780ab1e24e8f8c59",
          "blockLimit": "0x100",
          "blockNumber": "0x1",
          "chainId": "0x1",
          "extraData": "0x",
          "from": "0x57c7be32cbfb3bfed4fddc87efcc735b4e945fb3",
          "gas": "0x2faf080",
          "gasPrice": "0xa",
          "groupId": "0x1",
          "hash": "0x3961fac263d8e640b148ddcfafd71d2069e93a006abc937c32fb16cfa96e661d",
          "input": "0x4ed3885e0000000000000000000000000000000000000000000000000000000000000020000000000000000000000000000000000000000000000000000000000000000000000000000000000000000a464953434f2042434f53000000000000000000000000000000000000000000000000000000000000",
          "nonce": "0x3eb675ec791c2d19858c91d0046821c27d815e2e9c15160491220500000029688",
          "signature": {
            "r": "0x9edf7c0cb63645442aff11323916d51ec5440de979950747c0189f338afdcefd",
            "s": "0x2f3473184513c6a3516e066ea98b7cfb55a79481c9db98e658dd016c37f03dcf",
            "signature": "0x9edf7c0cb63645442aff11323916d51ec5440de979950747c0189f338afdcefd2f3473184513c6a3516e066ea98b7cfb55a79481c9db98e658dd016c37f03dcf00",
            "v": "0x0"
          },
          "to": "0x8c17cf316c1063ab6c89df875e96c9f0f5b2f744",
          "transactionIndex": "0x0",
          "value": "0x0"

```
            }
        ],
        "transactionsRoot":
 "0xb880b08df3b43a9ffc334d7a526522b33e004ef95403d61d76454b6085b9b2f
 1"
 }
```


4.1.2 武汉链


```
{
    "jsonrpc": "2.0",
    "id": 1,
    "result": {
        "difficulty": "0x2",
        "extraData":
"0xd883010a08846765746888676f312e31362e37856c696e75780000000000000000
0a7381c6f48e36d85b06ccc073de74836892b458b61caa512d5113e901abef0860
6aa9405d5ae035ff4e78daccfdc2362209d1f9475e1d77d50f018652372b69601",
        "gasLimit": "0x7a1200",
        "gasUsed": "0x3e21d",
        "hash":
"0x7bb27e906261ce362551463d34950ddf989b6d860eabc4664e831937108675a
d",
        "logsBloom":
"0x00000000000000000000000000000000000020080000000100000000010000000
00000000000000000400000004000000000000000000000000000000000000000000
80000000000000000000000000000000000000000000000000000000000100000400
00000000200000000000000000080000000000080000000000000000000100000000
00000000000000000000000000000000000100000002000000000800000000000000
00000000000000000000000000000000000800000000000000000000000000000000
00000000000000000000000000000000040000000100000000000000020000000080
000040000000000010000000000000000000000000000000002000000",
        "miner": "0x87a1243b8ac7363c3667d6ad6b89bec20ff022fe",
        "mixHash":
"0x0000000000000000000000000000000000000000000000000000000000000000
0",
        "nonce": "0x0000000000000000",
        "number": "0x1b5c4e",
        "parentHash":
"0xc0f3bc11897b020a9f71ea77cd7f3cdb7ef86778779ba17b3893eee0dc876b49",
        "receiptsRoot":
"0xe826d1a5608bd8351d5ee7ac9a10c3c8d08f3a0f703f687016e1edde8b469eb1
",
        "sha3Uncles":
```

"0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d4934
7",
"size": "0x475",
"stateRoot":
"0xd918ab8cd0e6dd07dda0a59db8d8866110f523d799e70f30bd0ac7201f3bca8
0",
"timestamp": "0x61c67c48",
"totalDifficulty": "0x36b89d",
"transactions": [
{
"blockHash":
"0x7bb27e906261ce362551463d34950ddf989b6d860eabc4664e831937108675a
d",
"blockNumber": "0x1b5c4e",
"from": "0x019ba4600e117f06e3726c0b100a2f10ec52339e",
"gas": "0x3e21d",
"gasPrice": "0x4d7c6d00",
"hash":
"0x25392b914c3c2d0fa7fd5fe41d2c218fe81a1be63eb80484c110432f40f8054f",
"input":
"0x146d9ddc0000000000000000000000009d37d92d3bca605a49f21642c309e57
8b16040fd00000000000000000000000000000000000000000000000000000000
00000060000000000000000000000000000000000000000000000000000000000
00000c00000000000000000000000000000000000000000000000000000000000
00000200000000000000000000000000000000000000000000000000000000000
00014000000000000000000000000000000000000000000000000000000000000
000a0000000000000000000000000000000000000000000000000000000000000
00200000000000000000000000000000000000000000000000000000000000000
40000000000000000000000000000000000000000000000000000000000000008
00000000000000000000000000000000000000000000000000000000000000005
6464632d320000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000564
64632d3100000000000000000000000000000000000000000000000000000000",
"nonce": "0x1a",
"to": "0x45bcf28556494fb116c4623f8f32091476933fe3",
"transactionIndex": "0x0",
"value": "0x0",
"type": "0x0",
"v": "0x2b89",
"r":
"0x6d0cb65392d3857e0ccb82f7c372b628074ec2d74bfde2230057cce97429aadd
",
"s":
"0x13cef22abd8d4e81eb875430706cd9cc3513900854d5474cd7a08b5751e1af97

"
                }
            ],
            "transactionsRoot":
"0xd7cd21ddeb2710404c7f075f5b66c3d626aad5819c757dac5b50e99007ba196
1",
            "uncles": []
        }
}

4.1.3 文昌链

{
    "jsonrpc": "2.0",
    "id": 1,
    "result": {
        "baseFeePerGas": "0x7",
        "difficulty": "0x0",
        "extraData": "0x",
        "gasLimit": "0xffffffff",
        "gasUsed": "0x0",
        "hash":
"0x8d24a1833846d4362cd039ed5ef8f762f3d4ed29b362fc8cb85f73ea0dfe499d",
        "logsBloom":
"0x00000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000",
        "miner": "0xc29d6bb924a12f890adaff4f04b5aba54cbfb6a5",
        "mixHash":
"0x00000000000000000000000000000000000000000000000000000000000000000000
0",
        "nonce": "0x0000000000000000",
        "number": "0xb44",
        "parentHash":
"0x28875d32ef3f2ff1b52e147459f3bd31a01b936a9d06ba8b47818886d9098ab9
",
        "receiptsRoot":
"0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
        "sha3Uncles":

"0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347",
"size": "0x253",
"stateRoot": "0xc7954e969d7583b0319fda81686007e7791fa858e26b09827010e1aed764b90f",
"timestamp": "0x61dd87c5",
"totalDifficulty": "0x0",
"transactions": [],
"transactionsRoot": "0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
"uncles": []
}
}

4.2 交易回执信息示例

4.2.1 泰安链

{
"blockHash": "0x977efec48c248ea4be87016446b40d7785d7b71b7d4e3aa0b103b9cf0f5fe19e",
"blockNumber": "0xa",
"contractAddress": "0x0000000000000000000000000000000000000000",
"from": "0xcdcce60801c0a2e6bb534322c32ae528b9dec8d2",
"gasUsed": "0x1fb8d",
"input": "0xb602109a0000000000000000000000000000000000000000000000000000000080000000000000000000000000000000000000000000000000000000000000000c0000000000000000000000000000000000000000000000000000000000000001000000000000000000000000000000000000000000000000000000000000001400000000000000000000000000000000000000000000000000000000000000020307831303030303030303030303030303030303030303030303030303030303030303030303030303000000000000000000000000000000000000000000000000000000008323031393037333000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000026161000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000262620000000000000000000000000000000000000000000000000000000000000000000000000",
"logs": [ ],

"logsBloom":
"0x00000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
00",
"output":
"0x00000000000000000000000000000000000000000000000000000000000000
00",

"root":"0x38723a2e5e8a17aa7950dc008209944e898f69a7bd10a23c839d341e9
35fd5ca",
"status": "0x0",
"to": "0x15538acd403ac1b2ff09083c70d04856b8c0bdfd",
"transactionHash":
"0x708b5781b62166bd86e543217be6cd954fd815fd192b9a124ee9327580df8f3
f",
"transactionIndex": "0x0"
}

4.2.2 武汉链

{
"jsonrpc": "2.0",
"id": 0,
"result": {
"blockHash":
"0x8cba08237576f10cc7728c63c4c1f0767613b8386c797e30ca295f58ab703325",
"blockNumber": "0x1b61ff",
"contractAddress": null,
"cumulativeGasUsed": "0x3161c",
"effectiveGasPrice": "0x4d7c6d00",
"from": "0x019ba4600e117f06e3726c0b100a2f10ec52339e",
"gasUsed": "0x3161c",
"logs": [
{
"address":
"0x920c8c4e79d975891765ec286afb8792b26a7887",
"topics": [

"0xe5e4f0955699fa27be3f7aeb5a9e4ab78b6a37914fd44e5b08c4b78e56f04aa0",

"0x000000000000000000000000019ba4600e117f06e3726c0b100a2f10ec52339e
",

"0x00000000000000000000000045bcf28556494fb116c4623f8f32091476933fe3"
],
"data":
"0xd3fc9864000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000a
",
"blockNumber": "0x1b61ff",
"transactionHash":
"0x31b09b1e618bcb8a513f4970300179f38583a881af4d03953bff79f1ca0e53d0",
"transactionIndex": "0x0",
"blockHash":
"0x8cba08237576f10cc7728c63c4c1f0767613b8386c797e30ca295f58ab703325",
"logIndex": "0x0",
"removed": false
},
{
"address": "0x45bcf28556494fb116c4623f8f32091476933fe3",
"topics": [

"0xc3d58168c5ae7397731d063d5bbf3d657854427343f4c083240f7aacaa2d0f62",

"0x000000000000000000000000019ba4600e117f06e3726c0b100a2f10ec52339e
",

"0x0000000000000000000000000000000000000000000000000000000000000000
0",

"0x000000000000000000000000019ba4600e117f06e3726c0b100a2f10ec52339e
"
],
"data":
"0x0000000000000000000000000000000000000000000000000000000000000000
b000000000000000000000000000000000000000000000000000000000000064
",
"blockNumber": "0x1b61ff",
"transactionHash":
"0x31b09b1e618bcb8a513f4970300179f38583a881af4d03953bff79f1ca0e53d0",
"transactionIndex": "0x0",
"blockHash":

"0x8cba08237576f10cc7728c63c4c1f0767613b8386c797e30ca295f58ab703325",
"logIndex": "0x1",
"removed": false
}
],
"logsBloom":
"0x00000000000000000000000000000000000000200800000000000000000001000000
00000000000000000400000004000000000000000000000000000000000000020000
80000000000000000000000000000000000000000000000000000000000100000400
00000000200000000000000000008000000000000000000000000000000001000000
00000000000000000000000000000000000000000000200000000000000000000000
00000000000000000000000000000000800000000000000000000000000000000000
00000000000000000000000000000000000000008100000000000000200000000
80000040000000000010000000000000000000000000000080000000000",
"status": "0x1",
"to": "0x45bcf28556494fb116c4623f8f32091476933fe3",
"transactionHash":
"0x31b09b1e618bcb8a513f4970300179f38583a881af4d03953bff79f1ca0e53d0",
"transactionIndex": "0x0",
"type": "0x0"
}
}

4.2.3 文昌链

{
"jsonrpc": "2.0",
"id": 0,
"result": {
"blockHash":
"0x9149b2138d340898c86b96ca07de62bd8612ccd66320bf4f45216b53e55b1ff
e",
"blockNumber": "0x700c",
"contractAddress": null,
"cumulativeGasUsed": "0x189aa",
"from": "0x2a14331f1f2d3ba0d750f4c4916e69b1dc38d721",
"gasUsed": "0x189aa",
"logs": [{
"address":
"0xdac50c90b934aded33b6adc9f5855ab8a9efb09a",
"topics":
["0x623400fa99cb4c5485584d19857bba72895e7e3400646f9fc46f627a605c267c
",

"0x000000000000000000000000005804a5f927ce7382ad194fd25bcaa189dad92a3
9"],
"data":
"0x00000000000000000000000000000000000000000000000000000000000000
e0000000000000000000000000000000000000000000000000000000000000001
2000000000000000000000000000000000000000000000000000000000000000
0100000000000000000000000000000000000000000000000000000000000001
6000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0100000000000000000000000000000000000000000000000000000000000001
a000000000000000000000000000000000000000000000000000000000000000
0831323331323332310000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0d706c6174666f726d4e616d65310000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
8313233313233323100000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0",
"blockNumber": "0x700c",
"transactionHash":
"0x4b286a32b43e3bf07e9b4967573871bf1eec808f33c577581c6f9f4620a3097e",
"transactionIndex": "0x0",
"blockHash":
"0x9149b2138d340898c86b96ca07de62bd8612ccd66320bf4f45216b53e55b1ff
e",
"logIndex": "0x0",
"removed": false
}, {
"address":
"0xdac50c90b934aded33b6adc9f5855ab8a9efb09a",
"topics":
["0x3a99d1f905eb66fd0850dd6e2156668b6ae1f16f41cd49dd7c6d4fb5147f784f
",
"0x000000000000000000000000005804a5f927ce7382ad194fd25bcaa189dad92a3
9"],
"data":
"0x00000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
01",
"blockNumber": "0x700c",
"transactionHash":
"0x4b286a32b43e3bf07e9b4967573871bf1eec808f33c577581c6f9f4620a3097e",
"transactionIndex": "0x0",

```
                "blockHash":
"0x9149b2138d340898c86b96ca07de62bd8612ccd66320bf4f45216b53e55b1ff
e",
                "logIndex": "0x1",
                "removed": false
            }],
            "logsBloom":
"0x00100000000000000000000000000000000000200000000000000000000000
00040000000800000000000000000000000000000000000000000000000000000
00000000000000000900000000000080000000000000000000000000000800000
00000000000002000000000000000000000000000000000000000000000000000
00000000000000000000000000000000200000000004000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000020000
00000000000000000000000000000000000000000000000000000000000000000
00",
            "status": "0x1",
            "to": "0xdac50c90b934aded33b6adc9f5855ab8a9efb09a",
            "transactionHash":
"0x4b286a32b43e3bf07e9b4967573871bf1eec808f33c577581c6f9f4620a3097e",
            "transactionIndex": "0x0",
            "type": "0x0"
        }
}
```


4.3 交易信息示例

4.3.1 泰安链


```
{
    "blockHash":
"0xfa639d1454362a8cdfcab1ca1948a5defaf7048b28f67e80780ab1e24e8f8c59",
    "blockLimit": "0x100",
    "blockNumber": "0x1",
    "chainId": "0x1",
    "extraData": "0x",
    "from": "0x57c7be32cbfb3bfed4fddc87efcc735b4e945fb3",
    "gas": "0x2faf080",
    "gasPrice": "0xa",
    "groupId": "0x1",
    "hash":
"0x3961fac263d8e640b148ddcfafd71d2069e93a006abc937c32fb16cfa96e661d",
```

```
        "input":
"0x4ed3885e000000000000000000000000000000000000000000000000000000000000000020000000000000000000000000000000000000000000000000000000000000000a464953434f2042434f53000000000000000000000000000000000000000000000000",
        "nonce":
"0x3eb675ec791c2d19858c91d0046821c27d815e2e9c15160491220500000029688",
        "signature": {
            "r":
"0x9edf7c0cb63645442aff11323916d51ec5440de979950747c0189f338afdcefd",
            "s":
"0x2f3473184513c6a3516e066ea98b7cfb55a79481c9db98e658dd016c37f03dcf",
            "signature":
"0x9edf7c0cb63645442aff11323916d51ec5440de979950747c0189f338afdcefd2f3473184513c6a3516e066ea98b7cfb55a79481c9db98e658dd016c37f03dcf00",
            "v": "0x0"
        },
        "to": "0x8c17cf316c1063ab6c89df875e96c9f0f5b2f744",
        "transactionIndex": "0x0",
        "value": "0x0"
    }
```


4.3.2 武汉链


```
{
        "jsonrpc": "2.0",
        "id": 1,
        "result": {
                "blockHash":
"0x8cba08237576f10cc7728c63c4c1f0767613b8386c797e30ca295f58ab703325",
                "blockNumber": "0x1b61ff",
                "from": "0x019ba4600e117f06e3726c0b100a2f10ec52339e",
                "gas": "0x3161c",
                "gasPrice": "0x4d7c6d00",
                "hash":
"0x31b09b1e618bcb8a513f4970300179f38583a881af4d03953bff79f1ca0e53d0",
                "input":
"0xd3fc9864000000000000000000000000019ba4600e117f06e3726c0b100a2f10ec52339e0000000000000000000000000000000000000000000000000000000000000064000000000000000000000000000000000000000000000000000000000000006000000000000000000000000000000000000000000000000000000000000012546f6b656e2d526564446174652d5778313600000000000000000000000000000000",
```

```
            "nonce": "0x29",
            "to": "0x45bcf28556494fb116c4623f8f32091476933fe3",
            "transactionIndex": "0x0",
            "value": "0x0",
            "type": "0x0",
            "v": "0x2b89",
            "r":
"0xac4f674255eb65589ad83e6c590c3c77a8e058ced78350be6bad2524aeed1f8b",
            "s":
"0x2dbab9a1c8f4b622dba70e9a14be27a53c736593ddbcaaf843be08d5be514b94"
        }
}
```


4.3.3 文昌链


```
{
    "jsonrpc": "2.0",
    "id": 2,
    "result": {
            "blockHash":
"0x9149b2138d340898c86b96ca07de62bd8612ccd66320bf4f45216b53e55b1ffe",
            "blockNumber": "0x700c",
            "from": "0x2a14331f1f2d3ba0d750f4c4916e69b1dc38d721",
            "gas": "0x493e0",
            "gasPrice": "0x989680",
            "hash":
"0x4b286a32b43e3bf07e9b4967573871bf1eec808f33c577581c6f9f4620a3097e",
            "input":
"0xbac7ad52000000000000000000000005804a5f927ce7382ad194fd25bcaa189d
ad92a390000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
1",
            "nonce": "0xfa",
            "to": "0xdac50c90b934aded33b6adc9f5855ab8a9efb09a",
            "transactionIndex": "0x0",
            "value": "0x0",
            "type": "0x0",
            "v": "0x1c",
            "r":
"0xcec487128c0d1b614984997152fe2d88e6963831bb902f6800f3355cdcdd12b",
            "s":
"0x6c3b007b5b54a05c0b55a5608cc8f5b5bff064ce160d95f61ad00539c51976df"
    }
}
```

4.4 离线生成账户

4.4.1 功能介绍

平台方或终端用户可以通过此方法生成离线账户

4.4.2 API 定义

- 方法定义：Account createAccount();
- 调用者：平台方、终端用户;
- 核心逻辑：无
- 输入参数:

字段名	字段	类型	必传	备注

- 输出参数:

字段名	字段	类型	必传	备注
账户地址	address	String	是	
公钥	publicKey	String	是	
私钥	privateKey	String	是	
助记词	mnemonic	String	是	

4.5 接入 Url 设置

4.4.1 功能介绍

平台方或终端用户可以通过此方法对接入的网关 URL 地址进行设置

4.4.2 API 定义

- 方法定义：Boolean setGatewayUrl(String gatewayUrl);
- 调用者：平台方、终端用户;
- 核心逻辑：无
- 输入参数:

字段名	字段	类型	必传	备注
网关 URL	gatewayUrl	String	是	

- 输出参数:

字段名	字段	类型	必传	备注
设置结果		Boolean	是	

4.6 接入 Key 设置

4.4.1 功能介绍

平台方或终端用户可以通过此方法对接入的 Key 进行设置，注：apiKey 在设置的时候 KEY 用“x-api-key”，值根据实际情况填写。

4.4.2 API 定义

- 方法定义：Boolean setGatewayApiKey(String apiKey);
- 调用者：平台方、终端用户;
- 核心逻辑：无
- 输入参数:

字段名	字段	类型	必传	备注

接入 KEY	apiKey	String	是	

- 输出参数:

字段名	字段	类型	必传	备注
设置结果		Boolean	是	