1. **Что такое JSON RPC?**

протокол удаленного вызова процедур, использующий формат JSON для передачи сообщений. Это очень простой протокол (очень похожий на XML-RPC), определяющий только несколько типов данных и команд. JSON-RPC поддерживает уведомления (информация, отправляемая на сервер, не требует ответа) и множественные вызовы.

1. **Опишите формат RPC-запроса.**

JSON-RPC работает, отсылая запросы к серверу, реализующему протокол. Клиентом обычно является программа, которой нужно вызвать метод на удалённой системе. Множество входных параметров может быть передано удалённому методу как массив или запись. Метод также может вернуть множество выходных данных (это зависит от реализации). Удалённый метод вызывается отправлением запроса на удалённый сервер посредством [HTTP](https://ru.wikipedia.org/wiki/HTTP) или [TCP/IP](https://ru.wikipedia.org/wiki/TCP/IP)-[сокета](https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D0%BA%D0%B5%D1%82_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D1%8B%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81)) (начиная с версии 2.0). При использовании HTTP [заголовок](https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%B3%D0%BE%D0%BB%D0%BE%D0%B2%D0%BA%D0%B8_HTTP) Content-Type определяется как application/json.

Все передаваемые данные — простые [записи](https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%BF%D0%B8%D1%81%D1%8C_(%D1%82%D0%B8%D0%BF_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)), [сериализованные](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B8%D0%B0%D0%BB%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) в JSON. Запрос — вызов определённого метода, предоставляемого удалённой системой. Он должен содержать три обязательных свойства:

* method — строка с именем вызываемого метода.
* params — массив данных, которые должны быть переданы методу, как параметры.
* id — значение любого типа, которое используется для установки соответствия между запросом и ответом.

Сервер должен отослать правильный ответ на каждый полученный запрос. Ответ должен содержать следующие свойства:

* result — данные, которые вернул метод. Если произошла ошибка во время выполнения метода, это свойство должно быть установлено в null.
* error — код ошибки, если произошла ошибка во время выполнения метода, иначе null.
* id — то же значение, что и в запросе, к которому относится данный ответ.

Для ситуаций, когда ответ не требуется, были введены уведомления. Уведомление отличается от запроса отсутствием свойства id, которое не требуется, так как не будет передан ответ. В таком случае свойство id может быть пропущено (версия 2.0) или установлено в null (версия 1.0).

1. **Позиционные параметры запроса.**

****

Передается массив, параметры можно доставать по номеру

1. **Именованные параметры запроса.**

****

Каждому параметру присваивается свое имя, по которому можно к нему обратиться

1. **Опишите формат RPC-ответа.**

Сервер должен отослать правильный ответ на каждый полученный запрос. Ответ должен содержать следующие свойства:

* result — данные, которые вернул метод. Если произошла ошибка во время выполнения метода, это свойство должно быть установлено в null.
* error — код ошибки, если произошла ошибка во время выполнения метода, иначе null.
* id — то же значение, что и в запросе, к которому относится данный ответ.

Для ситуаций, когда ответ не требуется, были введены уведомления. Уведомление отличается от запроса отсутствием свойства id, которое не требуется, так как не будет передан ответ. В таком случае свойство id может быть пропущено (версия 2.0) или установлено в null (версия 1.0).

1. **Что такое RPC-уведомление?**

Для ситуаций, когда ответ не требуется, были введены уведомления. Уведомление отличается от запроса отсутствием свойства id, которое не требуется, так как не будет передан ответ. В таком случае свойство id может быть пропущено (версия 2.0) или установлено в null (версия 1.0).

1. **Что такое пакет RPC-запросов?**

Чтобы отправить несколько объектов Запроса одновременно, Клиент МОЖЕТ отправить массив, заполненный объектами запроса.

Сервер должен ответить Массивом, содержащим соответствующие объекты ответа, после обработки всех объектов пакетного запроса. Объект ответа ДОЛЖЕН существовать для каждого объекта запроса, за исключением того, что НЕ ДОЛЖНО быть никаких объектов ответа для уведомлений. Сервер МОЖЕТ обрабатывать пакетный вызов rpc как набор параллельных задач, обрабатывая их в любом порядке и с любой шириной параллелизма.

Объекты ответа, возвращаемые из пакетного вызова, МОГУТ быть возвращены в любом порядке внутри массива. Клиент ДОЛЖЕН сопоставлять контексты между набором объектов запроса и результирующим набором объектов ответа на основе элемента id внутри каждого объекта.

Если сам пакетный вызов rpc не распознается как допустимый JSON или массив хотя бы с одним значением, ответ Сервера ДОЛЖЕН быть одним объектом ответа. Если в массиве ответов нет объектов ответа, которые должны быть отправлены клиенту, сервер НЕ ДОЛЖЕН возвращать пустой массив и вообще ничего не должен возвращать.