**Лабораторная работа 25**

**ПСКП, ПОИТ-3**

**Задание 01**

1. Разработайте JSONRPC-серверное приложение 25-01, выполняющие следующие удаленные процедуры.

|  |  |
| --- | --- |
| Процедура | Исполнение процедуры |
| sum(x1, x2, …, xn) | функция принимает переменное число параметров, вычисляет и возвращает сумму всех значений параметров |
| mul(x1, x2, …, xn) | функция принимает переменное число параметров, вычисляет и возвращает произведение всех значений параметров |
| div(x,y) | функция принимает 2 параметра: x и y, вычисляет и возвращает x/y |
| proc(x,y) | функция принимает 2 параметра: x и y, вычисляет и возвращает x/y\*100 |

1. Работоспособность сервера проверьте с помощью POSTMAN.

**Задание 02**. Ответьте на вопросы

1. Что такое JSON RPC?
2. Опишите формат RPC-запроса.
3. Позиционные параметры запроса.
4. Именованные параметры запроса.
5. Опишите формат RPC-ответа.
6. Что такое RPC-уведомление?
7. Что такое пакет RPC-запросов?
8. **Что такое JSON RPC?**

JSON-RPC (JSON Remote Procedure Call) — это легковесный протокол удаленного вызова процедур. Он позволяет программе вызывать функции или методы на удаленном сервере и получать результаты так, как если бы они выполнялись локально. Ключевые особенности: JSON для кодирования данных (все запросы и ответы передаются в формате JSON), транспортная независимость (хотя часто используется поверх HTTP, он может работать и с другими протоколами), простота (спецификация протокола очень проста и понятна), асинхронность (позволяет отправлять запросы и получать ответы асинхронно).

1. **Опишите формат RPC-запроса.**

JSON-RPC 2.0 запрос — это JSON-объект со следующими полями:  
 "jsonrpc" (обязательно): Строка, указывающая версию протокола JSON-RPC. Должна быть "2.0".

"method" (обязательно): Строка, содержащая имя вызываемого метода/процедуры на сервере (например, "sum", "div").  
 "params" (опционально): Структурированное значение (массив или объект), содержащее параметры для вызываемого метода. Если параметры не требуются, это поле может быть опущено.  
 "id" (обязательно, если ожидается ответ; отсутствует для уведомлений): Уникальный идентификатор, установленный клиентом. Это может быть строка, число или null. Сервер должен вернуть этот же "id" в своем ответе, чтобы клиент мог сопоставить ответ с запросом. Если "id" отсутствует, это уведомление.  
 Пример запроса: { "jsonrpc": "2.0", "method": "sum", "params": [1, 2, 3], "id": 1 }

1. **Позиционные параметры запроса.**

Если поле "params" в запросе является массивом JSON, то параметры считаются позиционными. Это означает, что параметры передаются методу в том порядке, в котором они указаны в массиве.  
 Пример: в запросе { "jsonrpc": "2.0", "method": "div", "params": [10, 2], "id": "div-request-001" }, 10 будет первым параметром (x), а 2 - вторым (y). В коде, params[0] будет 10, а params[1] будет 2.

1. **Именованные параметры запроса.**

Если поле "params" в запросе является объектом JSON, то параметры считаются именованными. Каждое свойство объекта "params" представляет собой параметр с соответствующим именем и значением. Порядок именованных параметров не имеет значения.  
 Пример: в запросе { "jsonrpc": "2.0", "method": "subtract", "params": { "minuend": 42, "subtrahend": 23 }, "id": "subtract-request-002" }, серверный метод "subtract" получил бы объект, где он мог бы получить доступ к params.minuend и params.subtrahend.

1. **Опишите формат RPC-ответа.**

JSON-RPC 2.0 ответ — это JSON-объект со следующими полями:  
 "jsonrpc" (обязательно): Строка, "2.0".  
 "result" (условно обязательно): Это поле должно присутствовать при успешном выполнении запроса и содержит результат, возвращенный методом. Если при выполнении метода произошла ошибка, это поле должно быть опущено.  
 "error" (условно обязательно): Это поле должно присутствовать, если при выполнении запроса произошла ошибка. Если ошибки не было, это поле должно быть опущено. Значением "error" является объект с полями "code" (числовой код ошибки), "message" (строка, описывающая ошибку) и опционально "data" (дополнительная информация об ошибке).  
 "id" (обязательно): Это поле должно совпадать со значением поля "id" из соответствующего запроса.  
 Одновременно поля "result" и "error" в ответе присутствовать не могут.  
 Пример успешного ответа: { "jsonrpc": "2.0", "result": 6, "id": 1 }  
 Пример ответа с ошибкой: { "jsonrpc": "2.0", "error": { "code": -32602, "message": "Invalid params: Expected an array of two numbers for div [x, y]." }, "id": "div-request-001" }

1. **Что такое RPC-уведомление?**

RPC-уведомление (Notification) — это особый тип JSON-RPC запроса, который не требует ответа от сервера. Клиент отправляет уведомление и не ожидает какого-либо подтверждения или результата. Уведомление отличается от обычного запроса тем, что у него отсутствует поле "id".  
 Пример уведомления: { "jsonrpc": "2.0", "method": "logEvent", "params": {"level": "info", "message": "User logged in"} }  
 Сервер, получив уведомление, обрабатывает его (если может) и не должен отправлять никакого ответа клиенту, даже в случае ошибки.

1. **Что такое пакет RPC-запросов?**

Пакет RPC-запросов (Batch request) — это возможность отправить несколько JSON-RPC запросов (или уведомлений) серверу за один раз, передав их в виде массива JSON-объектов.  
 Пример пакета: [ {"jsonrpc": "2.0", "method": "sum", "params": [1,2,4], "id": "1"}, {"jsonrpc": "2.0", "method": "notify\_hello", "params": [7]} ]  
 Сервер должен обработать каждый запрос в пакете и вернуть массив ответов. Для каждого запроса с "id" в массиве ответов должен быть соответствующий объект ответа. Для уведомлений (без "id") в массиве ответов не должно быть записей. Порядок ответов не обязан соответствовать порядку запросов, поэтому "id" важен для сопоставления. Если сам пакетный запрос некорректен, сервер возвращает один объект ошибки.

то такое WASM?

WASM (WebAssembly) - это бинарный формат инструкций для виртуальной машины. Он позволяет запускать код, написанный на языках вроде C, C++, Rust, в веб-браузерах (и других средах) с высокой производительностью, близкой к нативной. WASM предназначен для вычислительно сложных задач и дополняет JavaScript, а не заменяет его. Ключевые преимущества: производительность, портативность, возможность использования различных языков программирования и безопасность.

Что такое emcc?

emcc (Emscripten Compiler Frontend) - это основной инструмент командной строки (компилятор) из набора Emscripten SDK. Emscripten - это набор инструментов для компиляции кода, написанного на C и C++, в WebAssembly (WASM). Таким образом, emcc берет ваш C/C++ код и преобразует его в .wasm файл, который может быть выполнен в браузере или других средах, поддерживающих WASM. Он также часто создает JavaScript-файл для загрузки и взаимодействия с WASM-модулем.