Протокол взаимодействия терминального клиента и CLI сервиса

Терминальный клиент взаимодействует с CLI сервисом через соединение по сокету. Сообщения, которыми обмениваются терминальный клиент и CLI сервис являются обычными объектами языка Erlang. На стороне CLI сервиса для сериализации используется функция **term\_to\_binary/1**, а для десериализации - функция **binary\_to\_term/1**. На стороне терминального клиента для сериализации используется функция **erl\_encode** (из библиотеки **erl\_marshal**), а для десериализации — функция **erl\_decode** (из библиотеки **erl\_marshal**). Протокол взаимодействия выглядит следующим образом (в терминах объектов языка Erlang):

1) Выполнение некоторой команды на сервере:

Запрос на сервер: **{command, CommandLine}**, где **CommandLine** - данные, введенные пользователем, очищенные от пробельных символов с начала и с конца.

Возможные ответы с сервера:

* Данные для стандартного вывода **{command\_out, CommandOutput}**, где **CommandOutput** - данные с сервера, которые терминал выводит на стандартный вывод.
* Данные для стандартного вывода ошибок **{command\_err, CommandError}**, где **CommandError** - данные с сервера, которые терминал выводит на стандартный вывод ошибок.
* Окончание работы команды **{end, PromptStr}**, где **PromptStr** - строка подсказки (prompt).
* Окончание работы команды с последующим завершением клиентской сессии (и выполнения терминального клиента) **{stop, ""}**.
* Ошибка выполнения команды **{error, Reason}**, где **Reason** — причина ошибки.

2) Вход в систему (ввод аутентификационных данных пользователя):

Запрос на сервер: **{login, Username, Password}**, где **Username** — имя пользователя, **Password** — пароль пользователя в формате base64.

Возможные ответы с сервера:

* В случае успешного входа **{login\_success, Greeting}**, где **Greeting** — сообщение об успешном входе в систему.
* В случае неуспешного входа **{login\_fail, Reason}**, где **Reason** — причина неуспеха.
* В случае, если запрос не может быть обработан **{login\_error, Reason}**, где **Reason** — причина ошибки.

3) Выход из текущего режима работы:

Запрос на сервер: **{current\_mode\_exit}**.

Возможные ответы с сервера:

* В случае выхода из текущего режима работы **{current\_mode\_exit, Prompt}**, где **Prompt** - текущая строка подсказки (prompt).
* В случае завершения клиентской сессии (и терминального клиента) **{exit, ""}**.

4) Прерывание выполнения текущей выполняющейся команды:

Запрос на сервер: **{interrupt}**.

Ответ с сервера: ничего.

5) Запрос текущего состояния:

Запрос на сервер: **{current\_state\_request}**.

Ответ с сервера: **{current\_state\_response, Prompt}**, где **Prompt** - текущая строка подсказки (prompt).

6) Запрос списка расширений:

Запрос на сервер: **{****extension\_request, CommandLine}**, где **CommandLine** - данные, введенные пользователем, очищенные от пробельных символов с начала и с конца.

Ответ с сервера: **{extension\_response, CommonPrefix, ExtensionList}**, где **CommonPrefix** — общий префикс для списка расширений, **ExtensionList** - список расширений для данных, введенных пользователем. В случае, если списка расширений для данных пользователя нет, **CommonPrefix** будет пустой строкой, а **ExtensionList** — пустым списком.

7) Запрос помощи по команде

Запрос на сервер: **{help\_request, CommandLine}**, где **CommandLine** - данные, введенные пользователем, очищенные от пробельных символов с начала и с конца.

Ответ с сервера **{help\_response, Help}**, где **Help** — строка, содержащая помощь для команды. Если помощь для данной команды не доступна по той или иной причине, то будет возвращена пустая строка помощи.

8) Запрос списка доступных команд по префиксу

Запрос на сервер: **{suitable\_commands\_request, CommandLine}**, где **CommandLine** - данные, введенные пользователем, очищенные от пробельных символов с начала и с конца.

Ответ с сервера **{suitable\_commands\_response, CommandList}**, где **CommandList** — список доступных команд. Если доступных команд нет, то будет возвращен пустой список доступных команд.

9) Завершение работы:

Запрос на сервер: **{exit}**.

Ответ с сервера: ничего.

10) Уведомление о завершении работы клиентской сессии на CLI сервисе:

Запрос на сервер: ничего.

Ответ с сервера: **{exit, Reason}**, где **Reason** — причина завершения клиентской сессии.

Поведение:

* Результатом обработки запроса на выполнение команды на CLI сервисе будут несколько ответов **{command\_out, CommandOutput}** и/или **{command\_err, CommandError}** (несколько ответов означает, что будет прислано любое количество или 0). После завершения выполнения команды будет прислан ответ о завершении. Если после выполнения команды клиентская сессия должна быть завершена (также как и выполнение терминального клиента), то будет прислан ответ **{stop, ""}**. В противном случае — будет прислан ответ **{end, PromptStr}**. Если в процессе выполнения команды произойдут какие-либо ошибки, то в ответ на эти ошибки придут несколько ответов **{command\_err, CommandError}** и выполнение команды завершится.
* Запрос на выполнение команды на CLI сервисе является асинхронным для данного клиента; CLI сервис блокирует выполнение других команд, если на нем уже выполняется какая-либо команда. Если во время выполнения какой-либо команды, на CLI сервис придет еще один запрос на выполнение другой команды, то CLI сервис вернет ошибку выполнения команды **{error, Reason}**.
* Запрос на прерывание выполнения текущей команды **{interrupt}** является запросом без ответа. Результатом выполнения этого запроса является прерывание выполнения команды, если на CLI сервисе выполняется какая-либо команда. Результатом прерывания выполнения команды будет ответ **{end, PromptStr}** с CLI сервиса. Прерывание выполнения команды может произойти не сразу при получении данного запроса: вполне возможна ситуация, когда CLI сервис произведет некоторую очистку перед прерыванием выполнения команды.
* Запрос на вход в систему является синхронным. CLI сервис принимает данный запрос на обработку пока пользователь не войдет успешно в систему. После этого ответ на данный запрос возвращает ошибку обработки запроса.
* Результатом обработки запроса на вход в систему на CLI сервисе будет один из следующих ответов: **{login\_success, Greeting}**, **{login\_fail, Reason}** или **{login\_fail, Reason}**. Ответ **{login\_success, Greeting}** означает, что пользователь успешно вошел в систему и для него создана клиентская сессия; как результат, пользователь может работать с системой через данный экземпляр терминального клиента. Ответ **{login\_fail, Reason}** означает, что пользователю не удалось войти в систему (скорее всего из-за того, что он ввел неверное имя пользователя и/или пароль), но у пользователя есть еще минимум одна возможность ввести данные аутентификации через данный экземпляр терминального клиента. Ответ **{login\_error, Reason}** означает, что запрос на ввод данных аутентификации не может быть обработан (например, если последняя попытка ввода данных аутентификации не имела успеха и количество попыток ввода данных аутентификации достигло предельного значения); в этом случае пользователь не сможет работать с системой через данный экземпляр терминального клиента (и его выполнение будет завершено).
* Запрос на выход из текущего режима работы является синхронным. Данный запрос успешно обрабатывается только тогда, когда CLI сервис не занят выполнением какой-либо команды для данного клиента.
* Запрос текущего состояния является синхронным. Результат работы этого запроса не зависит от наличия или отсутствия выполняющейся команды на CLI сервисе для данного клиента.
* Запрос списка расширений является синхронным. Результат работы этого запроса не зависит от наличия или отсутствия выполняющейся команды на CLI сервисе для данного клиента.
* Запрос помощи по команде является синхронным. Результат работы этого запроса не зависит от наличия или отсутствия выполняющейся команды на CLI сервисе для данного клиента.
* Запрос списка доступных команд по префиксу является синхронным. Результат работы этого запроса не зависит от наличия или отсутствия выполняющейся команды на CLI сервисе для данного клиента.
* Запрос на завершение работы **{exit}** является запросом без ответа. Результат работы этого запроса не зависит от наличия или отсутствия выполняющейся команды на CLI сервисе для данного клиента.
* Уведомление о завершении клиентской сессии генерируется асинхронно (для терминального клиента) на стороне CLI сервиса. После генерации этого сообщения, на CLI сервисе завершает работу обработчик клиентских запросов и закрывается сокетное соединение для взаимодействия с клиентом (на стороне CLI сервиса). Терминальный клиент при получении данного сообщения выводит его на поток стандартного вывода и завершает свою работу.
* При получении уведомления о завершении клиентской сессии терминальный клиент в ответ может сгенерировать запрос на завершение работы **{exit}**. Будет ли в действительности генерироваться этот запрос зависит от реализации терминального клиента.