

[КАК СТАТЬ АВТОРОМ](#)[Технотекст](#)[Обучение ИБ: что с ним, на ваш взгляд, не...](#)**1124.55**

Рейтинг

MTC

Экосистема цифровых сервисов

**BosonBeard**

вчера в 11:56

«Слово из трёх букв», или Пишем SMS-аналог Wordle с помощью MTC Exolve



19 мин



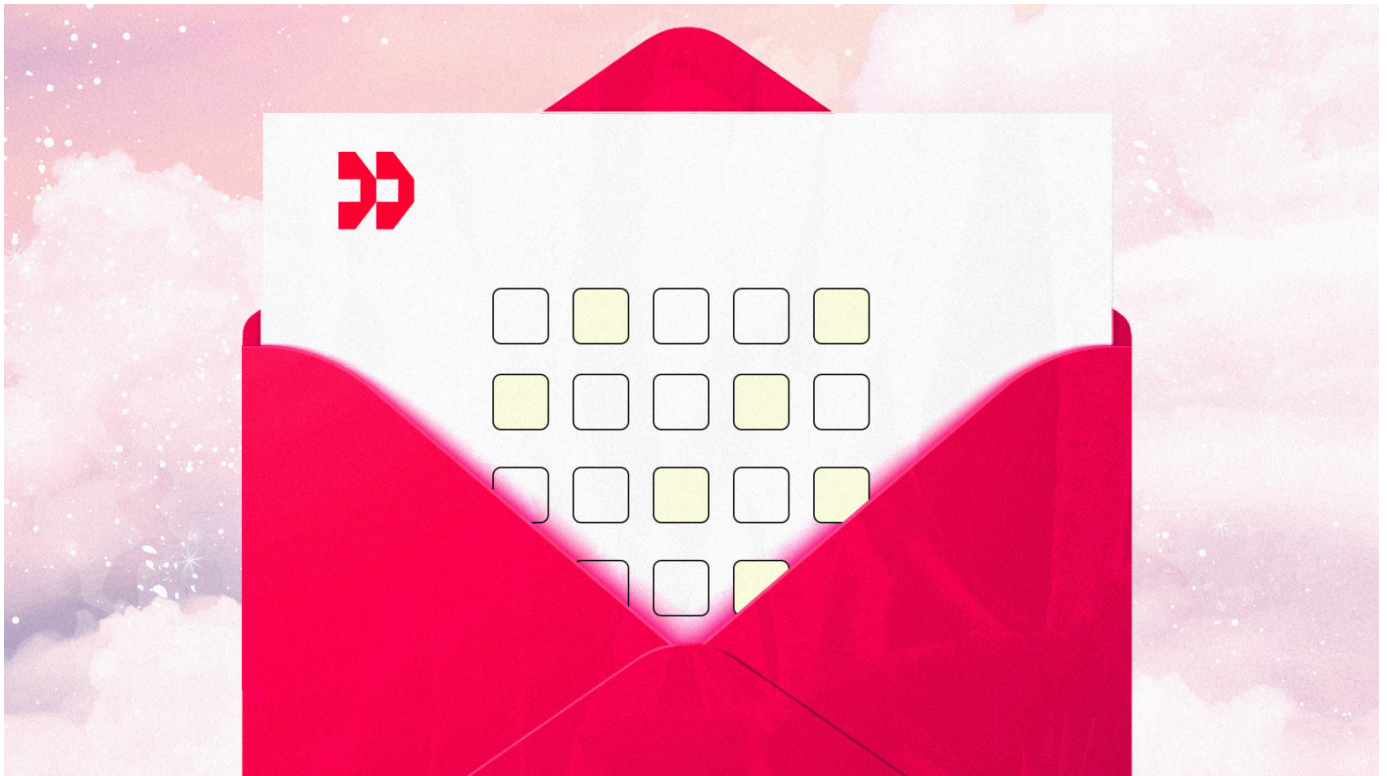
396

Блог компании MTC, IT-инфраструктура*, Облачные сервисы*, Сотовая связь

Привет, Хабр! Сегодня поговорим об игре «Угадай слово» и её вариациях, например Wordle («Вордли»). Скорее всего, вы с ней сталкивались, а если нет, поясню: она похожа на старую добрую «Виселицу» — есть загаданное слово, и надо за ограниченное количество попыток его угадать. Только отгадываем слово не по буквам, а целиком.

В какой-то момент игра стала так популярна, что некоторые компании интегрировали её в разные маркетинговые акции внутри приложений. Казалось бы, тема закрыта, можно расходиться. Как бы не так! Мне есть чем вас удивить, ведь сегодня займёмся разработкой именно такой игры, геймплей которой базируется на SMS-сообщениях. Подробности — под катом.





Отмечу, что благодаря тестовому периоду **MTC Exolve** можно попробовать себя в создании аналога Wordly. В нашей версии можно загадать определённое слово для кого-то из ваших друзей или близких.

И даже если у вас нет мобильного телефона или нулевой баланс счёта, статья может быть полезной. Ведь прежде чем дойти до SMS, напишем скрипт с HTTP API для игры. Поэтому сыграть в нашу «угадайку» можно и без SMS (и регистрации) с помощью привычных инструментов для запросов к API — например, cURL или Postman.

Записывайте рецепт. Для реализации понадобится:

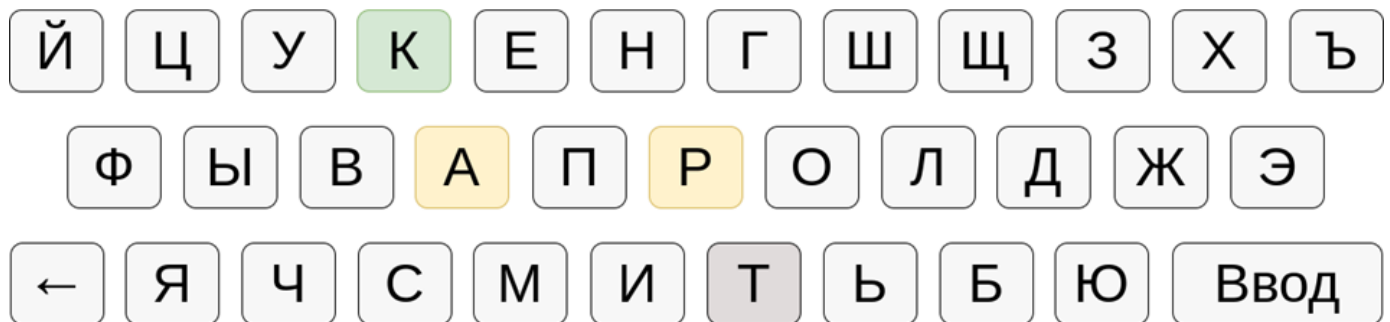
- любой хостинг с php и SQLite (можно бесплатный)
- оформить бесплатный тестовый период в **MTC Exolve** и подтвердить номер
- PHP-скрипт с API и скрипт обработки SMS

Что за «Угадайка» такая?

Если вдруг ажиотаж вокруг Wordle и его аналогов прошёл мимо вас, то объясню механику игры:

- есть загаданное слово — существительное из 5 букв
- слово, которое совершенно неизвестно игроку

- игрок вводит предполагаемое слово
- ему подсвечиваются буквы. В примере ниже:
 - серый — буквы нет в слове
 - жёлтый — буква есть в слове, но стоит не на своём месте
 - зелёный — буква указана верно
- также, как правило, нам для удобства подсвечены буквы, которые уже вводились
- игрок должен за 6 попыток отгадать слово



Пример среднестатистического интерфейса для игры в аналоги Wordly

Поскольку разрабатываем SMS-игру, то у нас есть некоторые ограничения на отображение информации, в том числе на количество символов в сообщении. Поэтому не будем полностью копировать механику игры. В конце концов это было бы неинтересно.

В нашей игре будут кое-какие отличия от оригинала:

1. Предоставляем возможность задать слово произвольной длины, через настройки, или просто возьмём случайное слово из словаря русских существительных (я брал словарь [здесь](#)).
2. Количество попыток равно длине слова.

3. Буквы, которые стоят не на своём месте в слове, подсвечивать не получится. Поэтому их записываем отдельным сообщением.
4. «Клавиатуры» с уже набранными буквами нет, их вполне заменит история SMS-переписки.

Скрин с демонстрацией игры я размещу в конце статьи. Ну а теперь, когда все точки над «і» расставлены, пришла пора браться за код.

Пишем API

Дисклеймер. Я не программист, поэтому не стоит принимать решения в этой статье за эталон кода, в процессе разработки я руководствовался принципом «главное, чтобы работало». Поэтому предложенный механизм с API вряд ли соответствует большинству хороших практик проектирования. Вы всегда можете его самостоятельно доработать. Совершенству нет предела.

Весь исходный код проекта размещён на [GitHub](#).

Ну а пока давайте разбираться со структурой нашего сервиса.

Методы API

Логика API реализуется внутри скрипта `api.php`.

API будет использовать 3 метода:

1. POST — для создания новой игры
2. PUT — для ввода предполагаемого слова
3. GET — для получения информации о текущем прогрессе

База данных

Данные будем хранить в базе данных SQLite3 (`db.sqlite`).

В таблице `games` будет информация о текущей игре.

В таблице `words` разместим словарь русских существительных.

Структура БД — на рисунке ниже:

Имя	Тип	Схема
Таблицы (3)		
games		CREATE TABLE "games" ("id" INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE, "phone" TEXT NOT NULL UNIQUE, "hidden_word"
id	INTEGER	"id" INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE
phone	TEXT	"phone" TEXT NOT NULL UNIQUE
hidden_word	TEXT	"hidden_word" TEXT NOT NULL
current_word	TEXT	"current_word" TEXT
last_try_word	TEXT	"last_try_word" TEXT
in_word	TEXT	"in_word" TEXT
attempts	INTEGER	"attempts" INTEGER DEFAULT 0
sqlite_sequence		CREATE TABLE sqlite_sequence(name,seq)
words		CREATE TABLE "words" ("word" TEXT)
word	TEXT	"word" TEXT

В таблице words всего один столбец со словами из словаря, что, в принципе, очевидно.

А вот для таблицы games расписать поля полезно:

- id — первичный ключ (ID записи)
- phone — телефон, для которого начата игра
- hidden_world — загаданное слово
- current_world — текущее состояние отгаданного слова, с указанием верно подставленных букв
- last_try_word — последнее отправленное слово, используется как защита от случайной повторной отправки того же самого слова (чтобы не списывать за это попытку)
- in_word — список букв, которые есть в слове, но стоят не на своём месте
- attempts — счётчик количества попыток

Вспомогательные файлы

Ещё наш API использует файл настроек game-config.json. В нём можно в формате «ключ-значение» загадать для телефона определённое слово.

Если номера нет в настройках, то слово будет взято из словаря.

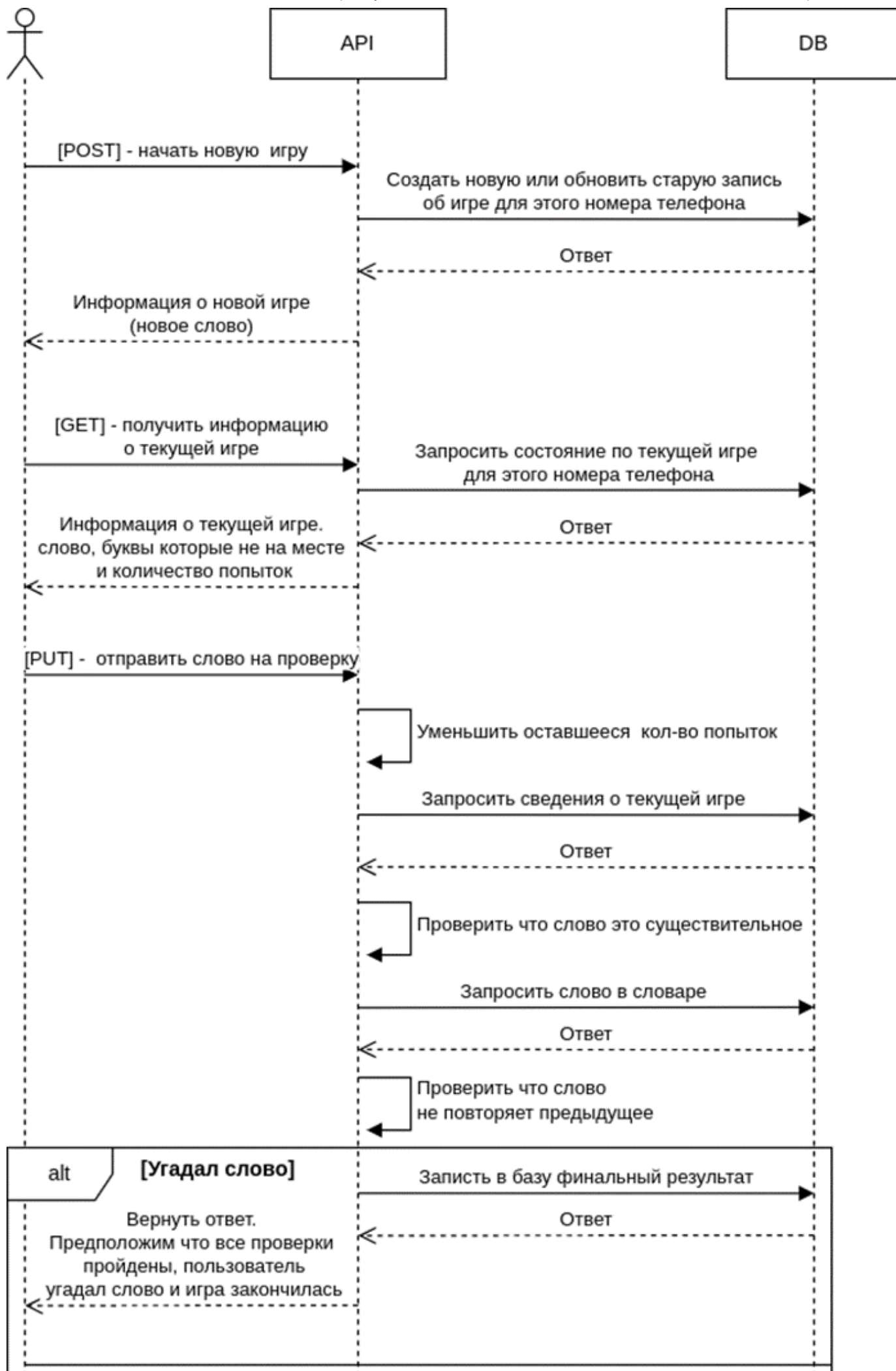
```
"phones_words": {
  "79121100234": "дом",
  "79111977999": "дерево"
},
```

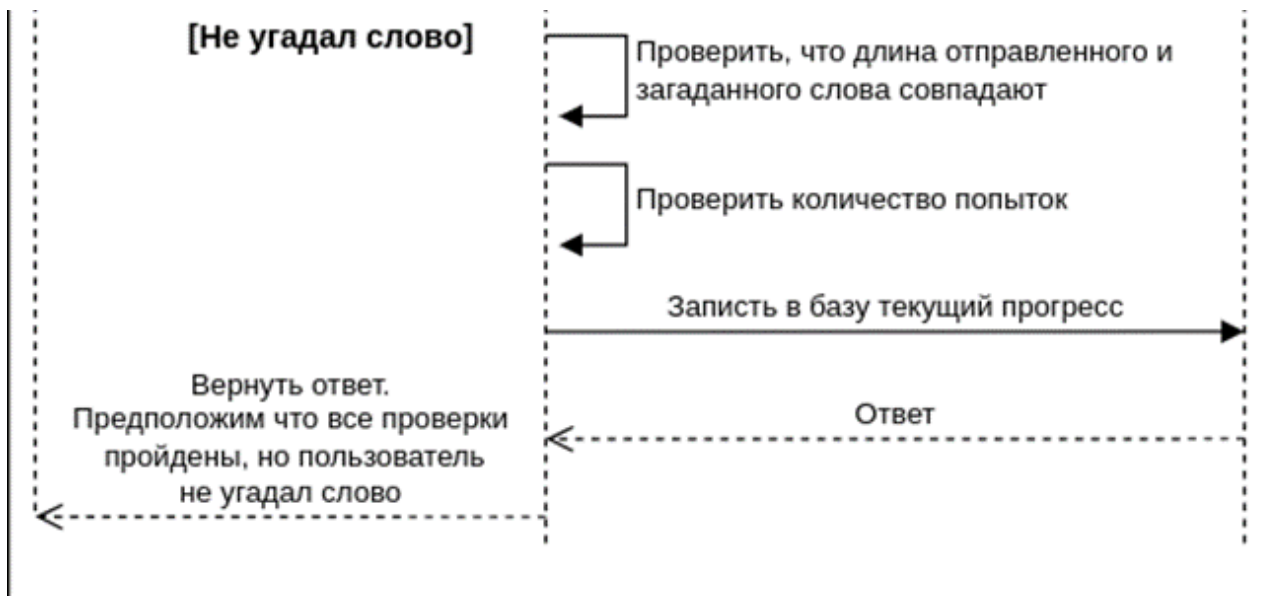
Ещё к вспомогательным файлам можно отнести файл логов api_log.txt. В логи пишутся телефон, слово (если есть), метод API и метка времени.

Пример: 79111977999 пелёнка POST 2023-09-12T00:07:53+03:00

Диаграмма последовательности

Давайте на диаграмме последовательности разберём подробно сценарий, в котором создаём новую игру, получаем информацию о текущем статусе и отправляем один раз слово на проверку.





Для упрощения схемы я опущу процесс чтения настроек из файла, записи логов и обработки исключений, сосредоточимся непосредственно на логике игры.

В этот раз вместо сценария предлагаю перейти к коду.

Код API

Полностью код файла `api.php` можно посмотреть под спойлером.

▸ [Код](#)

Давайте разберём основные блоки кода:

```
// Headers
header("Access-Control-Allow-Origin: *");
header("Access-Control-Allow-Methods: *");
header("Content-Type: text/plain");

// Get parameters from query
$method = $_SERVER['REQUEST_METHOD'];
$phone = $_REQUEST["phone"];
$user_word = mb_strtolower($_REQUEST["word"]);

// write request in log
$dt = date('c', time()); // get current time and date
$fw = fopen("api_log.txt", "a+");
fwrite($fw, $phone . " " . $user_word . " " . $method . " " . $dt . "\r\n");
fclose($fw);
```



```
//connect to DB
$db = new SQLite3('db.sqlite');

if ($phone) {
```

Стартовый блок.

Устанавливаем заголовки для страницы.

Затем считываем query-параметры. Чтобы упростить код, я решил даже для метода POST не использовать передачу параметров в body. Поэтому все параметры всегда передаются в query.

Далее добавляется запись в лог с текущей датой и временем. Потом подключаемся к SQLite-базе данных и в конце делаем проверку, указан ли в запросе телефон (если не указан, возвращается ошибка).

Потом — основная логика трёх методов API.

Начинаем с метода GET.

```
switch ($method) {
// GET method - game status info
case "GET":

// get last key command
$sql = "SELECT `hidden_word`,`current_word`,`in_word`,`attempts` FROM games WHERE `phone` = '$phone'";
// $sql = "SELECT `hidden_word`,`current_word`,`in_word` FROM games WHERE `phone` = '$phone'";
$result = $db->querySingle($sql, true);
$hidden_word = mb_strtolower($result['hidden_word']);
$current_word = mb_strtolower($result['current_word']);
$attempts = $result['attempts'];
$have_attempts = mb_strlen($hidden_word) - $attempts;

if ($result['in_word']) {
    $in_word = $result['in_word'] . " — не на своём месте \n";
}
// set API response

if ($result) {

if ($current_word != $hidden_word) {
```

```
$response =  
"Ваше слово: $result[current_word] \n" .  
"Осталось попыток: $have_attempts \n"  
. $in_word;  
} else {  
$response =  
"Вы угадали! \n" .  
"Загаданное слово: $hidden_word \n" .  
"Ещё оставалось попыток: $have_attempts \n";  
}  
} else {  
$response = "Начните новую игру";  
}  
  
break;
```

Как было показано на схеме выше, метод возвращает информацию о текущем ходе игры. Поэтому просто делаем SELECT-запрос в базу данных. Если в ответе БД столбец `in_word` был не пустой, записываем его значение в переменную по заданному шаблону сообщения, чтобы затем вернуть его пользователю.

Далее проверяем, есть ли вообще запись в БД (если нет, предложим начать новую игру). Если база данных нам что-то вернула, проверяем, угадал ли игрок слово ранее. В зависимости от этого возвращаем сообщение.

Следующий метод — POST.

```
case "POST":  
  
// get params from game-config.json  
$file = "game-config.json";  
$json = json_decode(file_get_contents($file), true);  
$hidden_word = mb_strtolower($json['phones_words'][$phone]);  
if (!$hidden_word) {  
$numRows = $db->querySingle("SELECT COUNT(*) as count FROM words");  
// echo "$numRows=".$numRows;  
$hidden_word = mb_strtolower($db->querySingle("SELECT * FROM words LIMIT 1 OFFSET " . ($numRows - 1)));  
}  
$current_word = str_repeat("_", mb_strlen($hidden_word));  
$have_attempts = mb_strlen($hidden_word);
```

```
// add new not processed command in DB
$sql = "UPDATE games
SET `hidden_word`='$hidden_word', `current_word`='$current_word',
`in_word`='', `attempts`=0, `last_try_word`='' WHERE `phone` = $phone";
$result = $db->querySingle($sql);

$sql = "INSERT INTO games (`phone`, `hidden_word`, `current_word`, `in_word`, `last_try_
VALUES('$phone', '$hidden_word', '$current_word', '', '')";
$result = $db->querySingle($sql);

// set API response
$response =
"Новая игра \n" .
"Слово: $current_word \n" .
"Букв: " . mb_strlen($current_word) . " \n";
"Осталось попыток: $have_attempts \n";
break;
```

В данном методе сначала читаем настройки из файла game-config.json. Для API нам надо получить привязку загаданного слова к номеру телефона. Если в настройках нет привязки, то вытаскиваем случайное слово из таблицы words БД.

Теперь необходимо создать в БД запись о новой игре. У нас для одного номера телефона создаётся только одна запись. Поскольку у меня на хостинге была старая версия SQLite, я не смог использовать UPSERT-логику, пришлось делить на вызов вначале UPDATE, а затем INSERT.

Если всё прошло успешно, возвращаем пользователю загаданное слово, количество букв в нём и количество оставшихся попыток.

► Остался самый большой метод — PUT.

В начале считываем из БД текущий прогресс.
Вычисляем новое количество оставшихся попыток.

Затем начинаются проверки:

1. Проверка на наличие записи в БД. Если записи нет, предложим начать новую игру.

2. Проверка, угадал ли игрок слово. Если угадал, обновим данные в БД, чтобы потом метод GET вернул информацию о победе. После чего пишем в переменную `$response` сообщение о победе. Если не угадал, идём дальше по цепочке проверок.
3. Проверяем, не отправил ли пользователь то же слово, что и в прошлый раз. Если отправил, то счётчик попыток не увеличиваем и сообщаем пользователю о повторе.
4. Проверяем, совпадает ли длина загаданного и отправленного слова. Если нет, сообщаем об этом пользователю, счётчик попыток не увеличиваем.
5. Проверяем, не закончились ли попытки.

После всех проверок вызывается функция `check_correct_letters` (описание будет ниже), записываем в базу текущий прогресс и возвращаем пользователю результат проверки.

В любом случае в конце всех кейсов пользователю возвращается ответ, ранее записанный в переменную `$response`.

```
        default:
            $response = '{"error":"unknown method"}';
            break;
    }
}
// phone is empty
else {
    $response = '{"error":"не передан телефон"}';
}

echo $response;
```

Осталось только разобрать функцию для проверки букв в слове.

```
function check_correct_letters($hidden_word, $user_word, $current_word)
{

    // get unicode array from string
    $hidden_word_arr = preg_split('//u', $hidden_word, 0, PREG_SPLIT_NO_EMPTY);
    $user_word_arr = preg_split('//u', $user_word, 0, PREG_SPLIT_NO_EMPTY);
    $current_word_arr = preg_split('//u', $current_word, 0, PREG_SPLIT_NO_EMPTY); //
    $in_word_arr = array(); // true letter on wrong place in word

    for ($i = 0; $i < count($user_word_arr); $i++) {
```

```
for ($j = 0; $j < count($hidden_word_arr); $j++) {
    if ($hidden_word_arr[$j] == $user_word_arr[$i]) {
        // get true letter on right place
        if ($j == $i) {
            if ($current_word_arr[$j] == "_") {
                $current_word_arr[$i] = $user_word_arr[$i];
            }
        }
        // get true letter on wrong place in word
        else {
            array_push($in_word_arr, $user_word_arr[$i]);
        }
    }
}

$in_word_arr = array_unique($in_word_arr, SORT_REGULAR);

return (object) [
    'current_word' => implode($current_word_arr),
    'in_word' => implode(",", $in_word_arr),
];
}
```

Передаём в функцию загаданное слово, отправленное пользователем слово и текущее слово в базе. Для начала разбиваем все строки на массивы букв. Затем создаём пустой массив букв, которые стоят в слове не на своих местах. Для каждого нового запроса это будет новый список.

Логика проверки незамысловатая. Существует основной цикл, в котором проходимся по каждому символу загаданного слова.

Если символы совпали, то проверяем, совпали ли их позиции в слове. Если совпали, значит буква угадана правильно, надо обновить текущее слово (`current_word`). Если позиции не совпали, значит буква есть в слове, но на другом месте, поэтому пишем её в `in_word_arr`.

Алгоритм неидеальный и иногда даёт небольшие сбои, но для нашей бесплатной игры вполне подойдёт. API готово, теперь можно протестировать его с помощью Postman.

POST ▼ [https://\[redacted\].ru/guess_word/api.php?phone=791\[redacted\]](https://[redacted].ru/guess_word/api.php?phone=791[redacted])

Params • Authorization Headers (7) Body Pre-request Script Tests Settings

Query Params

	Key	Value
<input checked="" type="checkbox"/>	phone	791[redacted]
<input type="checkbox"/>	word	пелёнка
	Key	Value

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize HTML ▼ ≡

```
1  Новая игра
2  Слово:  _____
3  Букв:  6
```

Пишем скрипт для обработки SMS

Теперь, когда вся логика игры реализована, подключаю обработку и отправку SMS-сообщений с помощью платформы MTC Exolve. В прошлой статье я уже писал, как зарегистрироваться и получить бесплатный номер телефона с тестовым балансом в 300 рублей, поэтому повторяться не буду. Но теперь нужно сделать дополнительный шаг в настройках личного кабинета и указать адрес нашего скрипта listener.php для обработки SMS. Перейдите в раздел «Настройки» и укажите URL до скрипта listener.php, размещённого на вашем хостинге.

habr

test

Настройки приложения

- Обзор
- API-ключи
- Номера
- SIP-аккаунты
- Callback
- Голосовые SMS
- Сообщения
- Аудиофайлы
- Статистика
- New Каналы
- Настройки

Общая информация


Приложение

habr

Описание

test

Уведомления о событиях

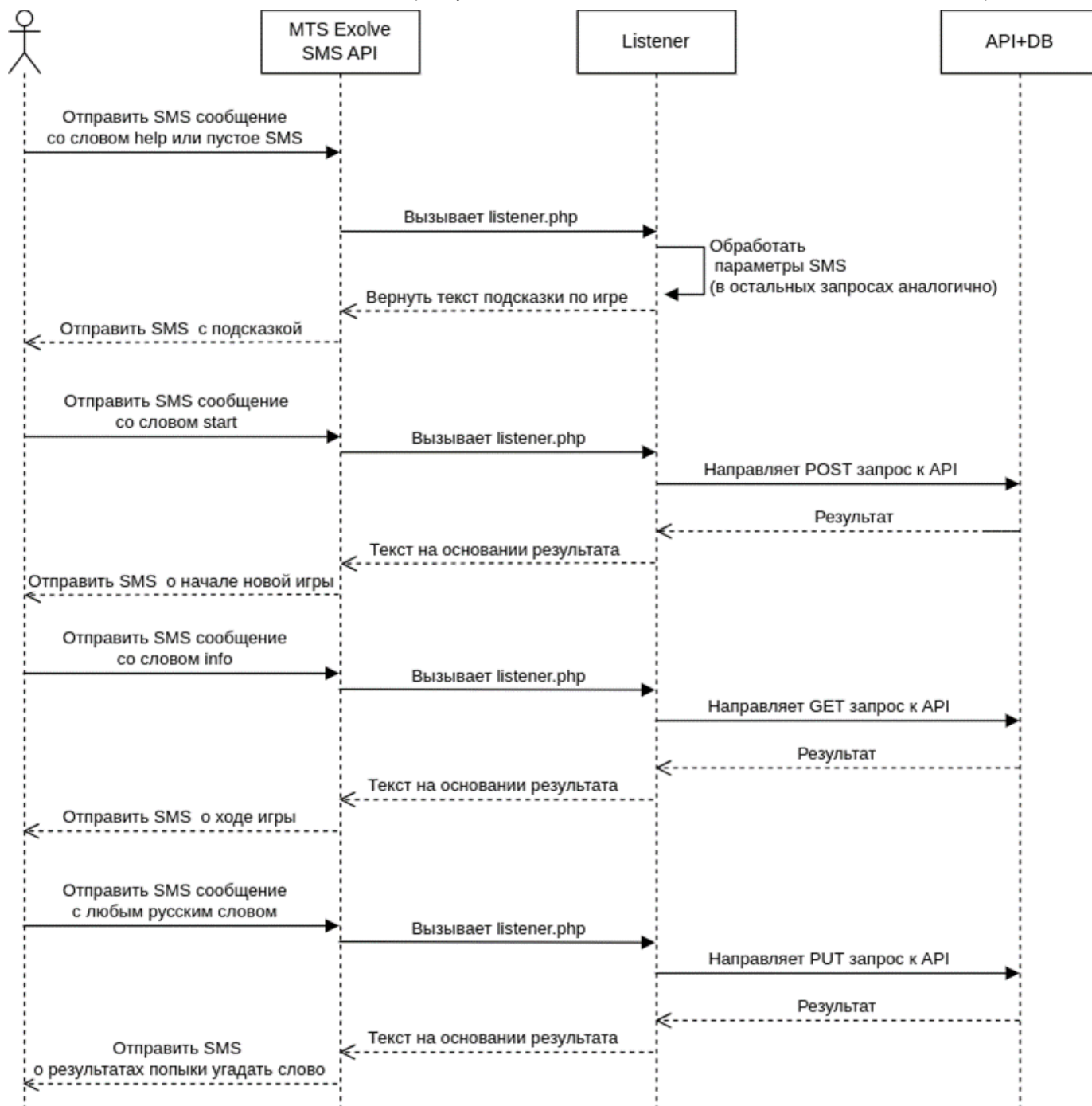
Добавьте URL-адрес , на который хотите получать уведомления по услугам в режиме онлайн. Так вы будете в курсе о ходе звонков и статусе сообщений

☐ Переадресация вхд. вызовов на URL☒ SMS[http://t\[REDACTED\]ru/guess_word/listener.php](http://t[REDACTED]ru/guess_word/listener.php)Хранение записей разговора ☐

После этого каждое пришедшее или отправленное SMS будет вызывать наш скрипт listener.php.

Второй скрипт будет выполнять достаточно простую задачу — получать уведомления об SMS-сообщениях. Обработать входящие сообщения в зависимости от текста, вызывать методы API и пересылать результат работы API в ответном SMS.

Всего будет 4 сценария для входящих SMS-сообщений, и все они представлены на диаграмме ниже:



Перейдём к коду. Полный код файла `listener.php` — под спойлером.

▸ [Код](#)

В данном блоке получаем тело уведомления от MTC Exolve. Подробнее с форматом уведомлений можно ознакомиться в документации.

Далее фильтруем только доставленные входящие сообщения и читаем вторую часть из файла настроек (объект `config`).

```
$file = "game-config.json";  
$json = json_decode(file_get_contents($file), true);  
$sms_api_key = $json['config']['sms_api_key'];  
$sms_api_url = $json['config']['sms_api_url'];  
$url = $json['config']['base_url'];
```

Содержимое файла настроек:

```
"config": {  
  "base_url": "ваш домен, на котором лежит скрипт/guess_word/api.php",  
  "sms_api_key": "Token приложения см. в ЛК MTC Exolve",  
  "sms_api_url": "https://api.exolve.ru/messaging/v1/SendSMS - адрес метода Exolve API"  
}
```

Затем в зависимости от текста сообщения:

- просто присылаем в ответном сообщении подсказку по игре (help или пусто)
- в остальных случаях вызываем методы api.php и возвращаем пользователю SMS с ответом

```
if ($text == "" or $text == "help") {  
  $response =  
  "Игра «Угадай слово», отправьте: \n" .  
  "start - новая игра \n" .  
  "info - текущий статус игры \n" .  
  "help или пусто - это сообщение \n" .  
  "любое другое слово - угадываемое слово \n";  
  $res = send_SMS($phone, $receiver, $response, $sms_api_key, $sms_api_url);  
}  
// start new game (use POST method game API)  
elseif ($text == "start") {  
  
  $res = call_API($url, "POST", $phone, $text);  
  $response = $res; // get text message from game api  
  $res2 = send_SMS($phone, $receiver, $response, $sms_api_key, $sms_api_url);  
}  
// send game status (GET method of game API)  
elseif ($text == "info") {
```

```
$res = call_API($url, "GET", $phone, $text);
$response = $res; // get text message from game api
$res2 = send_SMS($phone, $receiver, $response, $sms_api_key, $sms_api_url);
}
// send word for checking (PUT method of game API)
else {
$res = call_API($url, "PUT", $phone, $text);
$response = $res; // get text message from game api
$res2 = send_SMS($phone, $receiver, $response, $sms_api_key, $sms_api_url);
}

echo $response;
```

Функцию `call_API` подробно разбирать не будем — там просто вызов API с разными параметрами с помощью `cURL`. Тем более, что она по логике очень похожа на следующую функцию `send_SMS`.

```
function send_SMS($phone, $receiver, $text, $sms_api_key, $sms_api_url)
{
// create POST body
$data = array("destination" => $phone, "number" => $receiver, "text" => $text);
$data_json = json_encode($data);

// setup curl
$curl = curl_init();
curl_setopt($curl, CURLOPT_RETURNTRANSFER, 0);
curl_setopt($curl, CURLOPT_HTTPHEADER, array(
'Content-Type: application/json',
'Authorization: Bearer ' . $sms_api_key
));
curl_setopt($curl, CURLOPT_POST, 1);
curl_setopt($curl, CURLOPT_URL, $sms_api_url);
curl_setopt($curl, CURLOPT_POSTFIELDS, $data_json);
$result = curl_exec($curl);
sleep(1);
curl_close($curl);
return $result;
}
```

В данной функции настраиваются параметры отправки SMS-сообщения пользователю с помощью MTC Exolve. Параметры простые: кому отправить, с какого номера отправить и текст сообщения. Подробнее можно посмотреть в [документации](#).

Затем отправляем POST-запрос к API для отправки SMS-сообщений.

Безудержно развлекаемся

Когда всё готово, остаётся лишь одно — играть. Если не задавать слово заранее в настройках, то игра получается достаточно сложной, потому что в словаре очень много длинных и неожиданных слов. Но так даже лучше. Как и обещал, плоды наших трудов — на скриншотах ниже.



Единственный недостаток: в тестовом режиме SMS-сообщения отправляются только на номер, который привязан к личному кабинету. Поэтому если вы хотите сделать подарок-сюрприз кому-то в виде SMS-игры с каким-то своим необычным загаданным словом, рекомендую перевести аккаунт из тестового режима в полноценный.

Надеюсь, статья вызвала у вас интерес, буду рад конструктивным комментариям.

Теги: it-инфраструктура, облачные сервисы, сотовая связь, wordle, разработка, логические игры, разработка игр, игры и консоли, игры и игровые приставки


Хабы: [Блог компании MTC](#), [IT-инфраструктура](#), [Облачные сервисы](#), [Сотовая связь](#)

Редакторский дайджест

Присылаем лучшие статьи раз в месяц

✕

Электронпочта

 MTC

Экосистема цифровых сервисов



92

9

Карма Рейтинг


Роман @BosonBeard

Технический писатель / системный аналитик

 Комментарии 3

Публикации

ЛУЧШИЕ ЗА СУТКИ ПОХОЖИЕ



Bright_Translate

2 часа назад

На что способен самодельный очиститель воздуха, который можно собрать за 30 секунд?


 Простой  7 мин  2.9K

[Обзор](#) [Перевод](#)

 +31

 21

 6



MaFrance351

7 часов назад

Оживляем автоинформатор из подмосковного автобуса

 Простой  9 мин  2.2K

[Обзор](#) +29 9 18**Van4iniy**

19 часов назад

ChromeOS: почему я отказал своей мечте

 7 мин 14K[Из песочницы](#) +27 28 37**oldadmin**

6 часов назад

Будни техпода. Ошибки при подключении по RDP

 Простой 8 мин 2.9K[Тutorial](#) +23 23 8**it_union**

9 часов назад

Внимание. Ведется аудиоконтроль

 2 мин 2.6K +20 9 40**SLY_G**

20 часов назад

Раскрыта причина, по которой насекомые кружат вокруг огней по ночам

 5 мин 5.8K[Перевод](#) +18 23 14**LukaSafonov**

21 час назад

Web Application and API Protection (WAAP): эволюция WAF (Web Application Firewall)

 Средний  5 мин  1.6K

Аналитика

 +18

 21

 2



AlexanderKas1505

4 часа назад

Синдром айтишника 2

 Простой  5 мин  5.6K

Из песочницы

 +15

 23

 45



PatientZero

8 часов назад

Поисковый движок в 80 строках Python

 11 мин  2.3K

Тutorial

Перевод

 +15

 44

 0



Ilya12c

5 часов назад

Роман Тезиков про CV-проекты и промт-инжиниринг как базовый навык каждого человека

 7 мин  461

Интервью

 +12

 17

 1

Показать еще

ВАКАНСИИ КОМПАНИИ «МТС»

Старший UX редактор [Travel]

МТС · Москва · Можно удаленно

UX редактор [Умный дом]

МТС · Москва

Senior Android разработчик (Automotive) [МТС Авто]

МТС · Москва · Можно удаленно

Middle системный аналитик [Команда внедрения и поддержки]

МТС · Можно удаленно

Middle Android разработчик [МТС Авто]

МТС · Москва · Можно удаленно

Больше вакансий на Хабр Карьере

ИНФОРМАЦИЯ

Сайт	www.mts.ru
Дата регистрации	16 августа 2016
Дата основания	16 апреля 1993
Численность	свыше 10 000 человек
Местоположение	Россия



ССЫЛКИ

- Вакансии тут
career.habr.com
- Олимпиада True Tech Champ
truetechchamp.ru
- Больше о компании
mts-digital.ru

БЛОГ НА ХАБРЕ

5 часов назад

Акко MOD007B-HE PC: механическая клавиатура с магнитными свитчами. Небольшой обзор интересного аксессуара

 1.5K  6

вчера в 20:00

Как межзвёздные скитальцы «рассказывают» учёным об экзопланетах. Пример астероида Оумуамуа и кометы 2I/Борисова

1.5K 7

вчера в 15:00

Поломалось — ремонтируй и давай гарантию: в ЕС расширили положения «права на ремонт». Что изменилось?

2K 16

вчера в 11:56

«Слово из трёх букв», или Пишем SMS-аналог Wordle с помощью MTC Exolve

396 3

7 фев в 14:55

Поживём — увидим. Изучение средней продолжительности жизни людей даёт надежду на долголетие

2.6K 23

Ваш аккаунт	Разделы	Информация	Услуги
Войти	Статьи	Устройство сайта	Корпоративный блог
Регистрация	Новости	Для авторов	Медийная реклама
	Хабы	Для компаний	Нативные проекты
	Компании	Документы	Образовательные
	Авторы	Соглашение	программы
	Песочница	Конфиденциальность	Стартапам



Настройка языка

Техническая поддержка

