Connecting with JustPay
METHOD 1 (Code Pack)

Code generation: **by JustPay**
Code verification: **by Partner**

Table of Contents

1. Step-by-step guide for creating own service

1. Register with JustPay service and wait for email confirming your registration.
2. After you sign-in, define the commands and numbers of SMS Premium.
    a. "Commands" section – "add new command" button;
    b. Choose prefix from the list and a command suffix;
    c. Accept with "OK" button (the system shall automatically go to section 3).
3. Add the number to the command ("add number" link in the "commands" section)
    a. Choose the SMS Premium number from the list of available numbers
    b. Choose the generation method – in this case "Method 1"
    c. Choose the length and number of codes in a pack (default values are possible)
    d. Define the return SMS – write the contents of return SMS in the window; the series of '$1' signs shall be replaced by one of the codes from the pack
    e. Accept the number with "OK" button.
4. Having defined all necessary numbers (more than one can be added), order the activation of the command.
5. When the command is accepted by the system administrator, the user shall be informed via email. Awaiting the acceptance, further steps may be performed.
6. Download the code pack. In the "Codes" section find the list of generated code packs (for each command/SMS Number combination – there is one pack). The current (active) pack may be downloaded by clicking on "Download" in the "Current Pack" column (an own code pack may be added).
7. When 90% of the codes of the current pack are used, the Partner shall be informed via email about such fact. Thereafter, a new pack should be downloaded, awaiting in JustPay service and inserted like the previous pack.
8. When two packs are connected on the Partner's side, JustPay shall automatically download codes from the next pack once the first pack is exploited.

When JustPay system receives the User's SMS:

1. Justpay shall send back to the User a code from the active code pack located in the JustPay service (an identical pack is stored with the Partner).
2. The User enters the code on the Partner's website.
3. Partner verifies whether the code corresponds with the code pack earlier downloaded from JustPay service.
4. If the code is correct, the User shall be granted access to the chargeable section of the service.
5. Partner shall mark the User's codes as used in the Partner's database.

2. How to setup and implement example or individual scripts?

1. Preparation of the code tables
   a. Establish a relevant scheme in the database (the accomplishment of this step depends on what is made available on the hosting server where the website is to be launched; often the www panel is made available for database administration purposes).

   When the scheme is developed, $dsn variable should be modified accordingly in the example scripts.

   b. Create a code table – e.g. with an inquiry:

```
create table kody_codes (
        code varchar(20) not null,
        session_start timestamp,
        valid_secs integer not null,
        used boolean not null,
        primary key (code)
);
```

   c. Load code pack. For every code of the pack, perform SQL inquiry:

```
insert into kody_codes(code, valid_secs)
values ('kod', 3600);
```

   In the place of "code" enter the code from the downloaded file and in the place of 3600 insert the time (in seconds) during which the code is to be valid. This value may depend on the price of SMS – for instance codes for 1 zloty may be valid for one hour (as in the example), codes for 2 zlotys – 3 hours.

2. Prepare a non-secured page with an element allowing for entry into the secured section. In a relevant place on the page, add the following PHP code:

```
<?php
        include_once('secure/CodeCheck.php');
        $c = new CodeCheck();
        $c->render("proszę podać kod:", "secure/other.php");
?>
```

   In the place of "please provide code", insert the text to be displayed in the code window and instead of "secure/other.php", the secured page to which the viewer is to be redirected upon entering the correct code.

3. Preparation of a secured page. In order to secure the page, add at the very beginning in PHP code the following:

```
<?php
        session_start();
        include_once('CodeCheck.php');
        $c = new CodeCheck();
        if (!$c->isSessionCodeValid()) {
                include_once('enter.php');
                exit;
        }
?>
```

4. Example scripts. For method 1 the following example scripts are prepared (detailed description in section 5 below):
    a. Admin catalogue contains recreate.php script facilitating the creation of the code table and upload.php for filling in the table with codes.
    b. secure + index.php catalogue – example page (secured and non-secured page).

3. Method of code generation

The method of using the codes as well as validation of correctness and various types of security are on the Partner's side. This document aims to facilitate, by providing several examples, the launch of an own WWW service with a section secured with justpay codes. In the further parts of this document, the following assumptions are made:

1. Partner with "supergry" name completed registration with justpay system and as accepted by the administrator;
2. "Supergry" has a service with games available on www pages;
3. Part of the game functionality (more advanced options) are to be chargeable (made accessible only for users that paid a fee);
4. Access to chargeable sections is granted for a particular time.

4. Definition of commands

Justpay system allows for defining concurrently many commands for many numbers, whereupon Partners can launch more than one service on one account in Justpay. Such distribution is valuable when later analysing reports – income generated by respective services can be distinguished. Another way of using various commands is to distinguish the publication channels – one service can be promoted by advertisements in the Internet and at the same time by ads in the press. Creating two commands in such case shall help evaluate the effectiveness of respective channels.

Creating many commands has not greater impact on the technical aspects of the solution. If many commands promote one service, all received access codes should function in the same way. If there a several independent services, the code received by sending a command related to A service is simple incorrect when attempting to access B service. For such reason an assumption is made that the "Supergry" Partner has defined only one command: KOD.GRA.

It is absolutely different in case of SMS Premium numbers: if KOD.GRA is published on many numbers – for example: 7136 (number for 1 zloty), 7555 (number for 5 zloty) and 7936 (number for 9 zloty), the received code shall provide other rights. These rights shall depend on the specificity of the service and sold data, however in case of games xxx it is reasonable to condition the time of access to the page on the price of SMS. For instance – 24h for 7136, one week for 7555 and one month for 7936.

Having defined the command as described so far, the "Supergry" Partner should have 3 code packs (uploaded from CODE section) with names similar to the following ones:

- KOD.GRA_7136_20080416_1116.csv (command KOD.GRA for 7136),
- KOD.GRA_7555_20080416_1116.csv (command KOD.GRA for 7555),
- KOD.GRA_7936_20080416_1116.csv (command KOD.GRA for 7936)

5. Example scripts

The presented scripts are just examples – often very simplified or schematic. If integrated with the production system, please make sure to check them in terms of requirements of a particular application and compliance with the safety standards

Choice of technology

The most popular combination of technologies on generally available hosting servers is PHP + MySQL, therefore we decided to present them in such technology. When creating the examples, utmost care was taken to ensure that the code is most comprehensible, therefore transferring the solution to another technology should not pose problems.

WWW server configuration

The scripts were tested on PHP 5.2.0 with the following packs installed:

- PEAR 1.4.11,
- MDB2_Driver_mysql 1.4.1,
- MDB2 2.4.1.

MySQL: 5.0.32 server version — version of the database server is often quoted just for information purposes since the applied data structures and inquiries are simple enough to operate with any version. It was assumed in the presented examples that MySQL server operates on the same computer on which PHP scripts are triggered.

Database scheme

The following configuration is used in the examples: user codes, password: codes1234, scheme name: test_kody. Only one table is necessary for the functioning of the "securing" script: kody_codes. It contains the following fields:

- code varchar(20) — the field contains the access code sent via email
- session_start timestamp — the field contains the time of the first use of the code/time of uploading the code to the database
- valid_secs integer — the field contains the number of seconds during which the code is valid (since its first use)
- used boolean — the field is put as true upon first use of the code.

As access codes are unique, the code field may (and is) the master key of kody_codes table.

Once the described table is created, it should be provided with data from the code packs. The field valid_secs should contain a value corresponding to the number relating to a particular pack, e.g. for 7136, it shall be: 86400 (number of seconds during a 24h period).

In the examples of admin catalogue, the following scripts for creating and inputting data to the table are included:

- recreate.php — a script deleting and creating a new kody_codes table,
- upload.php — a simple form for inputting data into kody_codes table,
- doUpload.php — a script triggered by upload.php.

If the aforementioned scripts are used, please remember about the change of parameters of connection with the database. It is recommended to delete recreate.php script (or protect it in another way – if run accidentally, it shall result in loss of all data). Also, please remember about securing the admin catalogue against unauthorized access (e.g. with the use of .htaccess file or by its total removal – it is necessary only when adding a new code pack).

CodeCheck Class

CodeCheck class defined in CodeCheck.php file in secure catalogue is designed to secure PHP scripts with the use of codes. Connection with the database is initiated in the constructor of the class – and it is where the parameters of the connection should be configured.

For proper functioning of the class information is necessary stored in a session; if it is impossible to store SID identification in cookies, a different method of providing that parameter ought to be secured.

Using CodeCheck class is very simple. At the beginning of PHP script, which is to be secured, it is necessary to add the following code fragment:

```
<?php
    session_start();
    include_once('CodeCheck.php');
    $c = new CodeCheck();
    if (!$c->isSessionCodeValid()) {
        // akcja w przypadku niepoprawnego kodu [incorrect code]
        exit;
    }
?>
```

To display the form for entering the code by the user, the following code fragment may be used:

```
<?php
```

```
include_once('CodeCheck.php');
$c = new CodeCheck();
$c->render("podpis:", "strona.php");
?>
```

In the “podpis”/signature/ field, enter the etiquette that is to be associated with the text field for the code, whereas in the “strona.php” field – enter the page to which the user shall be redirected upon entering the correct code.

6. Example of a secured page

A complete example of a “secured” page comprises the following files:

- index.php — a non-secured page from which there are two ways of going to the secured version,
- secure/CodeCheck.php[1] — the abovementioned definition of CodeCheck class,
- secure/enter.php — an example of an enter code form,
- secure/index.php — the first secured page (from where it is possible to go to the second secured page)
- secure/other.php — the second secured page (allows for returning to the first: “master” secured page).

If the setup is correct, the example page should operate in the following manner:

1. without entering any code – it should be possible to view (via the browser) the first page (index.php);
2. any attempt to go to the “secured” page should result in the display of secure/enter.php page;
3. if a correct code is entered on index.php or secure/enter.php page, the user should be able to move between the secured and non-secured pages (without the need to enter the code once again);
4. the validity of the code lapses upon end of valid_secs from the moment when the code was entered for the first time – thereafter the code is considered incorrect and does not allow for entering secured pages.

7. Protection of non-text data

The presented example page illustrates how PHP scripts are secured. If it is necessary to protect non-text data (images, music files etc.), the presented solution shall not work. The problem may be solved in many ways. The simplest way is to introduce an indirect script and to transfer data to a catalogue to which WWW server has no access.

In secure catalogue, there is jpeg.php script, which illustrates how images such as JPEG can be secured. To use it, replace elements such as:

---

[1]Notation `katalog/plik.php` means: a file named `plik.php` is stored in `katalog` catalogue.

```
<img alt="podpis" src="dir/obrazek.jpg"/>
```

with elements such as:

```
<img alt="podpis" src="secure/jpeg.php?f=dir/obrazek.jpg"/>
```

In jpeg.php script only the catalogue needs to be configured (variable $BASE), where files with images are stored.

Should you have any questions, please contact us at: justpay@avantis.pl