**ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ**

Государственное образовательное учреждение высшего профессионального образования

**“Санкт-Петербургский государственный электротехнический университет “ЛЭТИ”**

**им. В. И. Ульянова (Ленина)” (СПбГЭТУ)**

|  |
| --- |
|  |

Факультет КТИ УТВЕРЖДАЮ

Кафедра АСОИУ Зав. кафедрой Советов Б.Я.

Специальность 230102 «…….»………………………………

**ЗАДАНИЕ**

**на дипломное проектирование**

|  |
| --- |
| Студенту Шустрову Александру Борисовичу |
| Место дипломного проектирования СПбГЭТУ «ЛЭТИ» |
|  |
| 1. Тема дипломного проекта (работы) |
| Автоматизированная система управления для облуживания клиентов в поликлинике |
|  |
|  |
| 2. Назначение разработки |
| Автоматизация документооборота и организация электронной очереди в поликлинике |
|  |
| 3. Перечень подлежащих разработке разделов по теме |
| Определения |
| Введение |
| Анализ предметной области и разработка технического задания |
| Программная реализация разработанной системы |
| Тестирование и подготовка презентации системы |
| Экономическое обоснование |
| Охрана интеллектуальной собственности |
|  |
| 4. Состав технической документации проекта |
| Пояснительная записка к дипломному проекту |
| Презентация |
|  |
| 5. Консультанты ( с указанием относящихся к ним разделов) |
| Консультант по экономическому обоснованию: Марков М. А. |
| Консультант по охране интеллектуальной собственности: Берковская К.И. |
|  |

Дата выдачи задания Дата сдачи проекта (работы)

«……»………………. «……»……………….

Руководитель………………………………………… Степулёнок Д.О.

Консультант от кафедры…………………………….. Степулёнок Д.О.

Студент……………………………………………….. Шустров А.Б.

**ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ**

Государственное образовательное учреждение высшего профессионального образования

**“Санкт-Петербургский государственный электротехнический университет “ЛЭТИ”**

**им. В. И. Ульянова (Ленина)” (СПбГЭТУ)**

|  |
| --- |
|  |

Факультет      КТИ                УТВЕРЖДАЮ

Кафедра    АСОиУ               Зав. кафедрой     Советов Б.Я

Специальность     230102      «…….»………………………………

**КАЛЕНДАРНЫЙ ПЛАН**

**дипломного проектирования**

|  |
| --- |
| Студента**Шустрова Александра Борисовича** |
| Тема дипломного проекта (работы) |
| Автоматизированная система управления для облуживания клиентов в поликлинике |
|  |
| Срок дипломного проектирования: 4 месяца |
|  |

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование раздела | Срок выполнения | Отметка  о выполнении |
| 1.  2.  3.  4.  5. | Анализ предметной области и разработка технического задания  Тестирование и подготовка презентации системы  Внедрение и тестирование  Технико-экономическое обоснование  Охрана интеллектуальной собственности | 19.10.11  20.12.11  05.01.12  18.01.12  25.01.12 | Выполнено  Выполнено  Выполнено  Выполнено  Выполнено |

Руководитель                                     Степулёнок Д.О.

Консультант от кафедры                  Степулёнок Д.О.

Студент                                               Шустров А.Б.

**Abstract**

The goal of thisdiploma work–develop automated storage system for documentationand program code for machines with numerical control.

System use MySQL database, PHP programming language (system is based on Yii Framework), HTML markup language, Cascading Style Sheets tables, JavaScript language.

Оглавление

[Определения 8](#_Toc316990900)

[Введение 11](#_Toc316990901)

[1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И РАЗРАБОТКА ТЕХНИЧЕСКОГО ЗАДАНИЯ 14](#_Toc316990902)

[1.1 Описание деятельности поликлиники 14](#_Toc316990903)

[1.2 Используемые в настоящее время документы 16](#_Toc316990904)

[1.1 Описание (формализация) бизнес-процессов 20](#_Toc316990905)

[1.2 Сфера применения и пользователи разрабатываемой системы 20](#_Toc316990906)

[1.3 Техническое задание на дипломное проектирование 20](#_Toc316990907)

[2. РАЗРАБОТКА И ПРОГРАММНАЯ РЕАЛИЗАЦИЯ СИСТЕМЫ 22](#_Toc316990908)

[2.1 Выбор средств реализации системы 22](#_Toc316990909)

[Web-система.. 22](#_Toc316990910)

[Настольные приложения. 22](#_Toc316990911)

[Где будет применяться? 22](#_Toc316990912)

[2.2 Языки и средства разметки для создания web-сайтов 22](#_Toc316990913)

[2.3 Языки программирования для создания серверной части 24](#_Toc316990914)

[2.4 Системы управления базами данных 27](#_Toc316990915)

[1.4 Средства разработки 29](#_Toc316990916)

[2.5 30](#_Toc316990917)

[2.6 Установка локального web-сервера denwer 30](#_Toc316990918)

[2.7 Настройка dbForgeStudioдля разработки 32](#_Toc316990919)

[Рис. 6. Настройки подключения к серверу баз данных 32](#_Toc316990920)

[2.8 Проектирование Базы Данных 32](#_Toc316990921)

[2.9 Описание параметров библиотеки (интерфейса подключения). 53](#_Toc316990922)

[2.10 Реализация библиотеки 55](#_Toc316990923)

[2.3.1 Основной файл библиотеки protect.php 55](#_Toc316990924)

[2.3.2 Подключаемый модуль библиотеки sd\_captcha.php 61](#_Toc316990925)

[3. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ СИСТЕМЫ 66](#_Toc316990926)

[3.1 Вход в систему, авторизация и аутентификация 66](#_Toc316990927)

[3.2 Использование для защиты от автоматической отправки сообщений, XSS-атак. Примеры фильтрации параметров SQL-запросов. 68](#_Toc316990928)

[3.3 Тестирование капчи 70](#_Toc316990929)

[4. ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ 72](#_Toc316990930)

[4.1 Концепция технико-экономического обоснования 72](#_Toc316990931)

[4.2 Потребительские свойства 72](#_Toc316990932)

[4.3 Затраты на разработку 73](#_Toc316990933)

[4.4 Трудоемкость и календарный план 78](#_Toc316990934)

[4.5 Комплексная оценка эффективности 80](#_Toc316990935)

[4.6 Выводы 80](#_Toc316990936)

[5. ВОПРОСЫ ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ 81](#_Toc316990937)

[5.1 Общие положения 81](#_Toc316990938)

[Авторское право на программу для ЭВМ и базу данных 82](#_Toc316990939)

[Правообладание 84](#_Toc316990940)

[Передача исключительных прав на программу для ЭВМ и базу данных 85](#_Toc316990941)

[Нарушение прав на программу для ЭВМ и базу данных 86](#_Toc316990942)

[5.2 Комплект основных документов для официальной регистрации программы для ЭВМ 87](#_Toc316990943)

[СПИСОК ЛИТЕРАТУРЫ 99](#_Toc316990944)

[ПРИЛОЖЕНИЕ 100](#_Toc316990945)

Определения

**Интернет** - всемирная система объединённых компьютерных сетей, построенная на использовании семейства протоколов ТCP/IP и маршрутизации пакетов данных. WWW-сервис (World Wide Web - всемирная паутина) - это гипертекстовая, гипермедийная, распределенная, интегрирующая, глобальная информационно-поисковая система в Интернет.

**Web-сайт**  - это информация (представленная в определенном виде), которая располагается на web-сервере и имеет свое уникальное имя (адрес) в Интернет. Web-сайт представляет собой некоторый набор текстовой и графической информации, организованный в виде определенного количества web-страниц, связанных между собой гипертекстовыми ссылками.

**Клиент-сервер —** вычислительная или сетевая архитектура, в которой задания или сетевая нагрузка распределены между поставщиками услуг (сервисов), называемых серверами, и заказчиками услуг, называемых клиентами. Нередко клиенты и серверы взаимодействуют через компьютерную сеть и могут быть как различными физическими устройствами, так и программным обеспечением. Программный код, обеспечивающий интерактивность web-страниц, называется **сценарием (скриптом).**

**Алгоритм** – совокупность операций, выполняемых в строго установленном порядке, для решения поставленной задачи.

**Информация** - сведения (сообщения, данные) независимо от формы их представления;

**Доступ к информации** - ознакомление с информацией, ее обработка, в частности, копирование, модификация или уничтожение информации.

**Правила разграничения доступа ПРД** - совокупность правил, регламентирующих права доступа субъектов доступа к объектам доступа.

**Несанкционированный доступ к информации НСД -** доступ к информации, нарушающий правила разграничения доступа с использованием штатных средств, предоставляемых средствами вычислительной техники или автоматизированными системами.

**Защита от несанкционированного доступа** - предотвращение или существенное затруднение несанкционированного доступа.

**Идентификация (обычно логин)** - присвоение субъектам и объектам доступа идентификатора и (или) сравнение предъявляемого идентификатора с перечнем присвоенных идентификаторов.

**Аутентификация (обычно пароль)** - проверка принадлежности субъекту доступа предъявленного им идентификатора, подтверждение подлинности.

**Целостность информации** - способность средства вычислительной техники или автоматизированной системы обеспечивать неизменность информации в условиях случайного и (или) преднамеренного искажения (разрушения).

**Поликлиника** – многопрофильное или специализированное лечебно-профилактическое учреждение для оказания амбулаторной медицинской помощи больным на приёме и на дому. В данном дипломном проекте рассматривается типовая районная поликлиника.

**База данных** – представленная в объективной форме совокупность самостоятельных материалов, систематизированных таким образом, чтобы эти материалы могли быть найдены и обработаны с помощью электронной вычислительной машины. В данном дипломном проекте имеется в виду реляционная БД.

**Реляционная БД** – база данных, основанная на реляционной модели данных. Реляционная модель данных включает следующие компоненты:

* Структурный аспект: данные в базе данных представляют собой набор отношений (relations – отсюда и название). В этой модели хранятся сущности и связи между ними. Сущности хранятся в виде таблиц, связи – в виде ограничений на значения столбцов.
* Аспект целостности – отношения (таблицы) отвечают определенным условиям целостности.
* Аспект обработки (манипулирования) — РМД поддерживает операторы манипулирования отношениями (реляционная алгебра, реляционное исчисление). В данной работе для манипулирования отношениями используется диалект языка SQL – MySQL.

**Целостность базы данных** (database integrity) – соответствие имеющейся в базе данных информации её внутренней логике, структуре и всем явно заданным правилам. Каждое правило, налагающее некоторое ограничение на возможное состояние базы данных, называется ограничением целостности (integrity constraint).

**SQL** (англ. Structured Query Language – «язык структурированных запросов») – универсальный компьютерный язык, применяемый для создания, модификации и управления данными в реляционных базах данных. SQL основывается на исчислении кортежей.

**MySQL** – свободная система управления базами данных (СУБД), MySQL является собственностью компании Oracle Corporation и распространяется под GNU General Public License или под собственной коммерческой лицензией. Помимо этого разработчики создают функциональность по заказу лицензионных пользователей, именно благодаря такому заказу почти в самых ранних версиях появился механизм репликации. MySQL является решением для малых и средних приложений и является наиболее распространённой СУБД на хостингах.

Введение

На производстве, где используется современное оборудование, зачастую наблюдаются случаи неполной реализации потенциала доступных технологических ресурсов. Это влечет за собой потери эффективности использования высокотехнологичного оборудования.

На примере ОАО ”Кировский завод” можно увидеть утечки производственных мощностей из-за различных недочетов в процессах производства. Одним из таких случаев является отсутствие автоматизации при наладке станков и отсутствие какого-либо технического мониторинга за производительностью оборудования.

В данной дипломной работе рассматриваются немецкие металлообрабатывающие станки с числовым программным управлением (далее ЧПУ) ”Monforts”. Станки “Monforts” оснащены компьютерами, которые имеют возможность коммутировать между собой и с другими системами посредством TCP/IP соединения. Это дает возможность создать из станков с ЧПУ компьютерную сеть, во главе с сервером, на котором находилось бы программное обеспечение, позволяющее более быстро и эффективно настраивать станки.

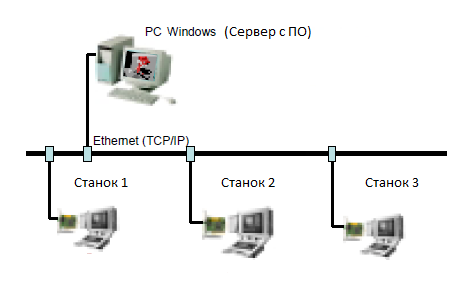


Рис. 1. Схема компьютерной сети

В настоящее время процесс наладки станка занимает много времени, вследствие того, что необходимая для производства той или иной детали документация не организованна и разбросана по производственному участку, а соответствующую этой документации программу приходится искать на разных станках.

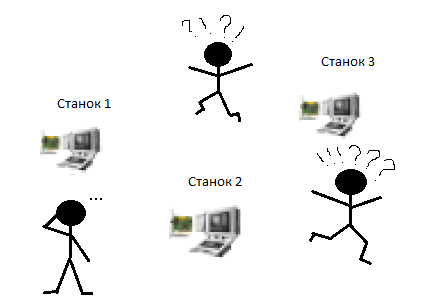


Рис. 2. Потеря эффективности производства

Большинство программ имеют несколько версий, т. к. могут быть предназначены для разных станков. Зачастую, практически невозможно определить принадлежность программы к тому или иному станку, тщательным образом не разобравшись в её коде, что ведет к большим затратам времени. Неверно запущенные на станках программы, ведут к возникновению ошибок в процессе наладки и вызывают довольно-таки весомый материальный ущерб. Неверно запущенная программа может привести к поломке дорогих элементов станка, сломанному инструменту и производству брака. Также, неправильно работающий станок является повышенным источником опасности для здоровья и жизни человека.

Вполне очевидно, что отсутствие какой-либо системы, которая обеспечивала бы порядок в процессе наладки, приводит к затрате большого количества времени и усилий, а также является причиной ошибок, которые могут привести к тяжелым последствиям. Это естественным образом вызывает потери в доходах того или иного предприятия, т.к. в процессе наладки станки не работают, а за допущенные из-за недостаточной производственной дисциплины ошибки приходится дорого платить.

Данная дипломная работа посвящена разработке автоматизированной системы для устранения описанных проблем.

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И РАЗРАБОТКА ТЕХНИЧЕСКОГО ЗАДАНИЯ
   1. Описание деятельности поликлиники

Для работы на станочном оборудовании, в особенности с ЧПУ, необходимо сначала настроить его, т. е. произвести *процесс наладки*. В настоящее время, процесс наладки станочного оборудования, прежде всего, включает в себяпоиск документации и нужной программы для производства детали. После того, как программа и соответствующая ей документация найдены, осуществляется загрузка программы в управляющую часть станка.

Под *управляющей частью* станка понимают компьютеризованную систему управления, управляющую приводами станков (центров), считывающую инструкции специализированного языкапрограммирования, который затем интерпретатором системы ЧПУ переводится из входного языка в команды управления главным приводом, приводами подач и контроллерами управления узлов станка.

Далее, после загрузки программы в управляющую часть, происходит последующая настройка инструмента. Например:

* определение типа и параметров зажима в патроне;
* измерение длины инструмента;
* привязка к нулям детали;
* определение положения инструмента в револьверной головке;
* определение положения револьверной головки относительно осей станка;

Настройка инструмента выполняется при помощи среды, установленной на управляющую часть станка. В данной работе рассматривается среда Siemens Sinumerik.

Siemens Sinumerik – полностью цифровая система для практически всех типов применений. Это системная платформа с прогрессивными функциями. Sinumerik представляет собоюсистему, которая подходит для сложных задач обработки и демонстрирует высокий уровень динамики и точности.Во всем мире Sinumerik применяется для: токарной обработки, сверления, фрезерования, шлифования, лазерной обработки, порезки, перфорации, изготовления оснастки и инструмента, как система управления прессами, для высокоскоростного раскроя материалов, обработки древесины и стекла, транспортировки, складских задач.

Sinumeric в модуле NCU (Numeric Control Unit — устройство числового управления) объединяет задачи ЧПУ, ПЛК и коммуникации. Установленный в каркас-носитель, NCU встраивается непосредственно в цифровую систему преобразования SIMODRIVE 611D, при этом он располагается справа, непосредственно у модуля питания-рекуперации.

Варианты процессоров NCU и системное программное обеспечение дает возможность оптимальной адаптации к станку и к задаче обработки. Такой модульный принцип позволяет оснастить целый ряд станков различного типа.

При помощи SINUMERIK 840D можно управлять максимум 31 осями/шпинделями. При максимальном использовании поддерживается до 10 каналов на каждую группу режимов работы и максимум 12 осей/шпинделей на каждый канал. Каждый канал может иметь свою собственную группу режимов работы.

SINUMERIK 840D позволяет просто и экономично обеспечить высокоэффективную защиту обслуживающего персонала и станков благодаря встроенным сертифицированным функциям защиты.

Все NCU изначально имеют встроенное подключение 4 быстрых цифровых входов/выходов ЧПУ.

Возможно объединение нескольких систем управления в одну.

После проведения настройки инструмента, наладчик приступает к пробной обработке детали. После того, как все эти этапы прошли успешно, по необходимости производится корректировка размеров и, станок становится полностью готовым к работе. На выполнение этих действий, как правило, затрачивается неопределенное количество времени, которое невозможно каким-либо образом предугадать. Это происходит вследствие некоторых причин, независящих от наладчика. Например:

* затрудненный поиск технологических процессов и программ для станков с ЧПУ
* отсутствие необходимого инструмента (кулачков, оправок, пластинок и пр.)
* наличие внешних факторов (отключение электричества, производственное собрание)
* неожиданная поломка оборудования, обеспечивающего работу станков (поломка гидравлической системы или конвейера для стружки)

Данный дипломный проект направлен, прежде всего, на устранение первого пункта из вышеизложенного списка. Для устранения данной проблемы необходимо создать базу данных с удобным интерфейсом, низкими системными требованиями и возможностью работы с ней из любого места, где есть подключение к интернету.

* 1. Используемые в настоящее время документы

Архитектура приложения состоит из трех слоев:

* Клиентский (Браузер)
* Серверный (Web-сервер)
* База данных

Рис. 3. Клиент-серверная архитектура системы

**Варианты реализации:**

* **Web-приложение**

Преимущества: не требуется установка (простота использования), централизованное обновление, обеспечение защиты, низкие требования к ресурсам компьютеров клиентов;современные технологии программирования постоянно развиваются в этом направлении, давая программистам все новые и новые инструменты по разработке именно web-ориентированных приложений;в обычных настольных приложениях чаще всего сервер базы данных не используется, а используется файловая модель хранения данных (например, в файлах Microsoft Access или DBF). Напротив, с большой долей вероятности, используя web-приложение, данные будут храниться в специальном SQL-сервере (так удобнее провайдерам). То есть надежность хранения данных у типичного web-приложения будет выше, чем у настольного приложения. Отсюда вытекает еще ряд достоинств использования web-приложения (из области сервисных задач): реализация разграничения доступа и резервного хранения информации обычно решена изначально в рамках типовых решений провайдера. Причем эти решения отработаны и оттестированы на многочисленных пользователях, что повышает степень доверия к ним и их надежность; легко модернизируема; можно получить доступ к нужным файлам из любого места, где есть компьютер с подключением к интернету.

Недостатки: скорость работы (для цели данной работы этот недостаток является несущественным), гибкость, затруднено использование внешних медиа-файлов (видео), требуется наличие постоянного соединения с интернетом.

* **Обычное настольное приложение**

Преимущества: возможность создания более мощного интерфейса.

Недостатки: массивность; низкая надежность хранения данных; для доступа к файлам, каждый раз необходимо будет устанавливать приложение туда, где оно будет использоваться (в данной работе этот недостаток является неприемлемым), низкая степень защиты информации;

* **Приложение для мобильных платформ**

Преимущества: малый размер; возможность использования такого приложения в любое время и в любом месте; легко в использовании;

Недостатки: низкая функциональность (недостаточная для реализации данного проекта); низкая степень надежности; отсутствие кроссплатформенности; неудобства средств управления приложением (отсутствие мыши, клавиатуры и пр.)

Для выполнения данного проекта, была выбрана реализация в виде Web-приложения. Основными предпосылками к этому стали:

* возможность использования такого приложения из любого места, где есть компьютер и подключение к интернету, т.к. зачастую то или иное производство разделено территориально (находится в разных цехах, производственных участках), но, при этом, имеет одни и те же задачи;

*К примеру: два одинаковых станка “а” и “б”, которые надо настроить на одну и ту же деталь располагаются в различных цехах: цехе ВХ 28 и в цехе №320, расстояние между которыми примерно равно 0.7 км. На станке “а” программа есть, а на станке “б” нет. Для того что бы настроить станок “б”, наладчику будет необходимо с каким-либо внешним накопителем преодолеть расстояние между цехами, скопировать программу, тем самым приостановив работу станка “а” и вернуться обратно. На выполнение данных действий он затратит от получаса времени. А при наличии доступа к единой базе данных это время сократится примерно в 6 раз.*

* простота использования, обусловленная тем, что web-приложение не требует установки;
* наличие централизованного сервера – это дает возможность объединить в единой базе данных необходимые для работы файлы с программами и документацией, структурировать их и осуществлять быстрый поиск, что позволяет экономить массу время и предотвратить возникновение ошибок, которые являются следствием “неразберихи” в процессе наладки оборудования.

Также, такие качества как надежность, низкие требования к компьютерным ресурсам, возможность модернизации, наличие централизованного обновления, высокая степень защиты и постоянное развитие в программировании Web-приложений, делают данный способ реализации актуальным и востребованным. Недостатки Web-приложений для выполнения данной работы являются несущественными.

* 1. Описание (формализация) бизнес-процессов
  2. Сфера применения и пользователи разрабатываемой системы
  3. Техническое задание на дипломное проектирование

Тема разработки: автоматизированная система хранения программного кода и документации для станков с ЧПУ.

Система будет реализована в виде web-системы, которая будет установлена на сервере локальной сети предприятия.

Все необходимые данные система будет хранить в базе данных MySQL. Для этого в базе данных должны быть таблицы для всех сущностей предметной области и связи между ними.

Серверная часть программы должна быть реализована на языке программирования PHP, интерфейс - на HTML с использованием языка программирования JavaScript, и каскадных таблиц стилей CSS.

Система должна позволять выполнять следующие действия:

1. Заходить в систему под своим логином и паролем. Если введённого сочетания логина и пароля нет в системе, должно выдаваться соответствующее сообщение об ошибке.
2. В систему должна быть встроена система контроля версий, т.е. для каждого файла в базе данных должен храниться набор версий.

Данная задача разбивается на следующие подзадачи:

1. Фильтрация полей:
   1. Фильтрация логина и пароля + обрезание количества символов.
   2. Фильтрация e-mail (адреса электронной почты).
   3. Общего текстового поля (textarea).
2. Фильтрация должна пресекать XSSи SQLInjectionатаки.
   1. Для защиты XSSвырезаются все теги
   2. Для защиты от SQL атак код приводится в «правильный вид»
3. При отправке формы для предотвращения подбора пароля и для защиты от спама в гостевой книгебудет применяться защита методом captcha. Капча подключается к библиотеке отдельным файлом **sd\_captcha.php**

Для использования модуля нужно в вызывающемPHP-скрипте присвоить принятые от пользователя значения специальным переменным, которые будут обработаны внутри библиотеки. Далее следует поместить подключение библиотеки (при помощи include).После выполнения библиотеки параметры будут отфильтрованы нужным образом (в соответствии с их типом).

1. РАЗРАБОТКА И ПРОГРАММНАЯ РЕАЛИЗАЦИЯ СИСТЕМЫ
2. Выбор средств реализации системы

Web-система..

Настольные приложения.

Где будет применяться?

1. Языки и средства разметки для создания web-сайтов

**HTML**, Hyper Text Markup Language (язык разметки гипертекста) — базовая технология интернета. HTML - стандартный язык разметки документов во Всемирной паутине. Большинство web-страницы создаются при помощи языка HTML (или XHTML). Язык HTML интерпретируется браузером и отображается в виде документа, в удобной для человека форме.

При помощи HTML можно создавать отдельные элементы, а в дальнейшем компоновать их между собой. HTML позволяет формировать текстовые блоки, форматировать их, выделяя абзацы, организовывать и создавать любые таблицы и списки. Есть возможность управлять цветовым оформлением документов, добавлять на сайт звуки, анимацию, картинки. HTML, естественно, помогает организовывать связь документов, разделов и страниц сайта между собой посредством гиперссылок.

Главный недостаток HTML — отсутствие средств, при помощи которых было бы возможно контролировать просмотр web-страниц независимо от пользователя. Web-мастер подготавливает страницу под какие-то определенные параметры, разрешение монитора, вид браузера и т.д. Пользователь же, изменив эти параметры у себя на компьютере, может получить на выходе страницу, существенно отличающуюся от замыслов программиста.

**XML**(eXtensibleMarkupLanguage, расширяемый язык разметки) - язык структурной разметки текстовой информации. XML предназначен для хранения структурированных данных (взамен существующих файлов баз данных), для обмена информацией между программами, а также для создания на его основе более специализированных языков разметки (например, XHTML), иногда называемых словарями. XML является упрощённым подмножеством языка SGML.

Главное преимущество XML заключается в том, что язык этот можно дописывать и расширять, добавляя новые тэги, создавая свои пользовательские XML-словари. По мнению аналитиков, XML постепенно должен стать основным стандартом во всемирной сети, включив в себя большинство других использующихся ныне форматов.

**CSS**. Технология CSS (CascadingStyleSheets, каскадные таблицы стилей) позволяет с помощью специального макроязыка задать форматирование страницы. В файле CSS мы можем задать параметры для всех тегов HTML по отдельности, а также для группы тегов, создать некий шаблон, данные которого будут в дальнейшем влиять на все элементы страницы.

Стили можно задать не только для одной конкретной страницы, но и для всего сайта в целом. Также технологию CSS можно использовать практически на любом сервере без каких-либо ограничений.

**DHTML** (DynamicHyperTextMarkupLanguage) – это способ создания интерактивного веб-сайта, использующий сочетание статичного языка разметки HTML, встраиваемого (и выполняемого на стороне клиента) скриптового языка JavaScript и CSS (каскадных таблиц стилей).

DHTML базируется на объектной модели документа DOM (DocumentObjectModel), технология эта встроена во все виды браузеров и помогает осуществить доступ к любому отдельно взятому элементу web-страницы (который в дальнейшем можно видоизменить). Технология DHTML позволяет нам осуществлять программное управление всеми тегами и атрибутами и создавать сложные пользовательские интерфейсы. DHTML позволит программисту спроектировать такие интерактивные элементы, как движущийся фон, расположенный под статичным текстовым содержимым страницы, любые сложные объекты, выпадающие меню, мигающие при наведении курсора кнопки, анимацию, титры и многое другое.

1. Языки программирования для создания серверной части

Все описываемые языки программирования относятся к числу объектно-ориентированных.

Из них Perl, РНР, Python, JAVA — языки, исполняющиеся на сервере (серверные языки). JavaScript — язык, исполняющийся на браузере (клиентский). Компьютер, открывающий доступ к собственным ресурсам, носит название **сервера**, а получающий такой доступ — **клиент**.

**Perl** (Practical Extrifiction and Report Language, практический язык выборки и составления отчетов) оптимизирован для обработки большого количества текстовых файлов. Язык Perl способен обрабатывать огромное количество документов на web-узле (сервере), многие серверы написаны как раз на языке Perl.

Web-мастеру Perl поможет проверить верность перекрестных ссылок на сайтах, осуществит проверку заполнения элементов формы, позволит присоединять базы данных с последующей их обработкой. Perl может работать и с протоколом FTP. Это позволяет автоматизировать получение файлов с других серверов, а в сочетании с возможностями языка по обработке текстовых файлов дает web-мастеру возможность создавать сложные информационные системы.

**PHP–** это скрипт-язык (scriptinglanguage), встраиваемый в HTML, который интерпретируется и выполняется на сервере.

Язык программирования PHP создавался специально для сети. PHP лучше всего охарактеризовать так: программист пишет сценарии — код, сервер, получая готовые указания, обрабатывает информацию и выводит результат на монитор пользователя.

PHP помогает создавать динамические web-приложения, такие как форумы, счетчики статистики, может работать с большими объемами данных — массивами. Очень хорошо язык PHP подходит для написания систем управления сайтами «движков».

Для того чтобы сайт быстрее загружался, стоит разбить страницу на отдельные элементы. Язык программирования PHP позволяет «на лету», за считанные доли секунды, объединять отдельные составляющие в цельную web-страницу.

Основное достоинство языка PHP — простота изучения. Он будет интуитивно понятен программистам, обладающим навыками в таких языках, как Perl и C, и для новичка вполне доступен.

В PHP нет поддержки ссылок на переменные, из-за чего сложно обрабатывать масштабные структуры данных, например, вложенные списки. Нет проверки типов данных передаваемых функций, что может повлечь за собой множественные ошибки.

Основное отличие от CGI-скриптов, написанных на других языках, типа Perl или C – это то, что в CGI-программах вы сами пишете выводимый HTML-код, а, используя PHP – вы встраиваете свою программу в готовую HTML-страницу, используя открывающий и закрывающий теги (в примере <?php и ?>).

Отличие PHP от JavaScript, состоит в том, что PHP-скрипт выполняется на сервере, а клиенту передается результат работы, тогда как в JavaScript-код полностью передается на клиентскую машину и только там выполняется.

**Python** прост для освоения, а между тем этот язык программирования на уровень выше, чем PHP и C. Это достигается за счет встроенных высокоуровневых структур данных (списки, массивы, словари).

Главное его достоинство то, что интерпретатор Python реализован практически на всех платформах и операционных системах.

**JAVA** представлен на сегодняшний момент в двух вариантах: JavaScript и собственно JAVA.

**JavaScript** — по сути, всего лишь надстройка стандарта HTML, которая значительно расширяет возможности документа, созданного в этом формате. Часть кода, написанная на JavaScript, внедряется в документ HTML как подпрограмма и вызывается на исполнение соответствующей строкой HTML-кода. Весь сценарий JavaScript умещается в одном-единственном теге <script>. Встроенный в браузер интерпретатор языка воспринимает и скрипт, и сам код гипертекста как единый документ, обрабатывая те и другие данные одновременно.

При помощи JavaScript можно полностью контролировать формат всплывающих окон и встроенных фреймов, организовывать такие интерактивные элементы, как часы, календарь, бегущие строки и иную анимацию, создать чат, сделать красивое меняющееся меню.

**JAVA** — это уже нечто большее: полноценный кроссплатформенный язык программирования, который будет правильно обработан и компьютером с операционной системой Windows, и ПК под управлением UNIX. Большинство web-камер, передающих на сайты изображение в реальном времени, работают на базе приложений JAVA.

**ASP** (Active Server Pages, активные страницы сервера) — это мощная технология от компании Microsoft, позволяющая легко разрабатывать интернет-приложения и служащая для создания динамических web-страниц. ASP нельзя назвать языком программирования, так как это только технология предварительной обработки, позволяющая подключать программные модули во время процесса формирования страницы. Ее можно сравнить с библиотекой для сети: «Большой энциклопедический словарь» с возможностью поиска по архивам. ASP имеет преимущество в скорости по сравнению с другими технологиями, основанными на скриптах.

Удобство ASP заключается в том, что она работает на специальном сервере PWS (Personal Web Server), который входит в состав операционной системы Windows.

1. Системы управления базами данных

**MySQL –** это одна из самых популярных и самых распространенных СУБД (система управления базами данных) в интернете. Она не предназначена для работы с большими объемами информации, но ее применение идеально для интернет-сайтов, как небольших, так и достаточно крупных.

MySQL отличатся хорошей скоростью работы, надежностью, гибкостью. Работа с ней, как правило, не вызывает больших трудностей. Поддержка сервера MySQL автоматически включается в поставку PHP.

Немаловажным фактором является ее бесплатность. MySQL распространяется на условиях общей лицензии GNU (GPL, GNU Public License).

Ранее для долговременного хранения информации мы работали с файлами: помещали в них некоторое количество строчек, а затем извлекали их для последующей работы. Задача длительного хранения информации очень часто встречается в программировании Web-приложений: подсчёт посетителей в счётчике, хранение сообщений в форуме, удалённое управление содержанием информации на сайте и т.д.

Между тем, профессиональные приёмы работы с файлами очень трудоёмки: необходимо заботится о помещении в них информации, о её сортировке, извлечении, при этом не нужно забывать, что все эти действия будут происходить на сервере хост-провайдера, где с очень большой вероятностью стоит один из вариантов Unix - следовательно, нужно так же заботится о правах доступа к файлам и их размещении. При этом объем, кода значительно возрастает, и совершить ошибку в программе очень просто.

Все эти проблемы решает использование базы данных. Базы данных сами заботятся о безопасности информации и её сортировке, и позволяют извлекать и размещать информацию при помощи одной строчки. Код с использованием базы данных получается более компактным, и отлаживать его гораздо легче. Кроме того, не нужно забывать и о скорости - выборка информации из базы данных происходит значительно быстрее, чем из файлов.

Приложение на РНР, использующее для хранения информации базу данных (в частности MySql) всегда работает быстрее приложения, построенного на файлах. Дело в том, что базы данных написаны на языке C++, и написать на PHP программу, которая работала бы с жёстким диском эффективнее базы данных - задача неразрешимая по определению, поскольку программы на PHP в принципе работают медленнее, чем программы на C++, так как РНР - интерпретатор, а С++ компилятор.

Таким образом, основное достоинство базы данных заключается в том, что она берёт на себя всю работу с жёстким диском и делает это очень эффективно.

* **Таблицы «ключ-значение» (хэш-таблицы)**

Это [структура данных](http://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), реализующая интерфейс [ассоциативного массива](http://ru.wikipedia.org/wiki/%D0%90%D1%81%D1%81%D0%BE%D1%86%D0%B8%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D1%8B%D0%B9_%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2), а именно, она позволяет хранить пары (ключ, значение) и выполнять три операции: операцию добавления новой пары, операцию поиска и операцию удаления пары по ключу. Существует два основных варианта хеш-таблиц: с цепочками и открытой адресацией. Хеш-таблица содержит некоторый массив H, элементы которого есть пары (хеш-таблица с открытой адресацией) или списки пар (хеш-таблица с цепочками).

Выполнение операции в хеш-таблице начинается с вычисления [хеш-функции](http://ru.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F) от ключа. Получающееся хеш-значение i = hash(key) играет роль индекса в массиве H. Затем выполняемая операция (добавление, удаление или поиск) перенаправляется объекту, который хранится в соответствующей ячейке массива H[i].

Ситуация, когда для различных ключей получается одно и то же хеш-значение, называется [коллизией](http://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BB%D0%BB%D0%B8%D0%B7%D0%B8%D1%8F_%D1%85%D0%B5%D1%88-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8). Такие события не так уж и редки — например, при вставке в хеш-таблицу размером 365 ячеек всего лишь 23-х элементов вероятность коллизии уже превысит 50 % (если каждый элемент может равновероятно попасть в любую ячейку) . Поэтому механизм разрешения коллизий — важная составляющая любой хеш-таблицы.

В некоторых специальных случаях удаётся избежать коллизий вообще. Например, если все ключи элементов известны заранее (или очень редко меняются), то для них можно найти некоторую [совершенную хеш-функцию](http://ru.wikipedia.org/w/index.php?title=%D0%A1%D0%BE%D0%B2%D0%B5%D1%80%D1%88%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F_%D1%85%D0%B5%D1%88-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F&action=edit&redlink=1), которая распределит их по ячейкам хеш-таблицы без коллизий. Хеш-таблицы, использующие подобные хеш-функции, не нуждаются в механизме разрешения коллизий, и называются хеш-таблицами с прямой адресацией.

Число хранимых элементов, делённое на размер массива H (число возможных значений хеш-функции), называется коэффициентом заполнения хеш-таблицы (load factor) и является важным параметром, от которого зависит среднее время выполнения операций.

* **Goggle App Engine**

Google App Engine позволяет создавать и размещать веб-приложения в тех же системах, в которых находятся приложения Google. В App Engine можно быстро и легко разработать, развернуть, администрировать и масштабировать приложения, не заботясь об аппаратных средствах, исправлениях и резервном копировании.

Разработанные в этой среде приложения являются зависимыми от компании “Google”, что является неприемлемым для данной работы.

* 1. Средства разработки

**Работа с базой данных MySQL**

dbForge Studio for MySQL

<http://www.devart.com/ru/dbforge/mysql/studio/download.html>

**HTTP-сервер для разработки**

Для разработки использовался локальный сервер из пакета Denwer (Джентльменский Набор Web-Разработчика). <http://www.denwer.ru/>

В пакет входят:

* HTTP-сервер Apache;
* Интерпретатор языка программирования PHP;
* Сервер баз данных MySQL.

**Средаweb-разработкидля PHP/HTML/CSS/JavaScript PHPStorm.**

<http://www.jetbrains.com/phpstorm/>

2. Установка локального web-сервера denwer

Запустите скачанный инсталлятор Денвера. Вы увидите перед собой нечто вроде следующего:

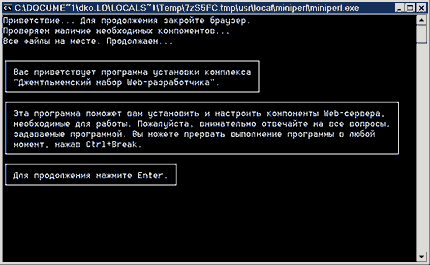


Рис. 4. Установка denwer

Вас спросят о том, в какой каталог вы хотели бы установить комплекс (по умолчанию используется C:\WebServers, вам нужно лишь нажать Enter, чтобы согласиться с этим выбором). В указанном каталоге будут расположены абсолютно все компоненты системы, и вне него никакие файлы в дальнейшем не создаются (исключая ярлыки на Рабочем столе).

Настоятельно рекомендуем вам устанавливать комплекс в каталог первого уровня — то есть, C:\WebServers, а не, например, C:\My\WebServers. Дело в том, что инсталляторы пакетов расширений ищут базовый комплект именно на первом уровне по всем дискам. И, если не находят, заставляют вас ввести имя директории вручную.

Далее вам предложат ввести имя виртуального диска, который будет связан с только что указанной директорией. Рекомендуем вам согласиться со значением по умолчанию (Z:). Важно, что диска с этим именем еще не должно содержаться в системе — чаще всего так и происходит с диском Z:.

После этого начнется копирование файлов дистрибутива, а под конец вам будет задан вопрос, как именно вы собираетесь запускать и останавливать комплекс. Удобнее «Создавать виртуальный диск при загрузке машины».

В некоторых версиях Windows 98 имеется ошибка, в результате которой диск с первого раза не отключается — мои эксперименты и исследования показали, что это именно ошибка в subst, а не что-то иное.

Собственно, вот и все. Вы сможете выполнить установку комплекса, имея всего 2 клавиши на клавиатуре: Enter и Y (чтобы ввести Yes).

После установки на рабочем столе появятся 3 иконки:



Рис. 5. Иконки для работы с denwer.

Для запуска локального web-сервера разработки надо нажать на «StartDenwer».

1. Настройка dbForgeStudioдля разработки

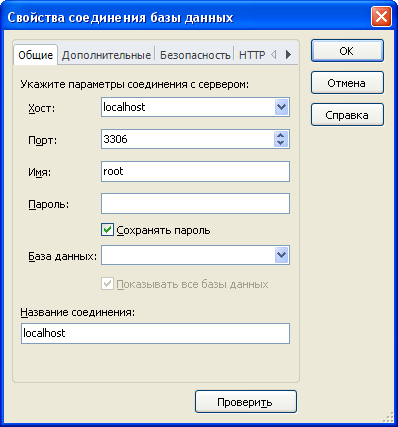


Рис. 6. Настройки подключения к серверу баз данных

1. Проектирование Базы Данных

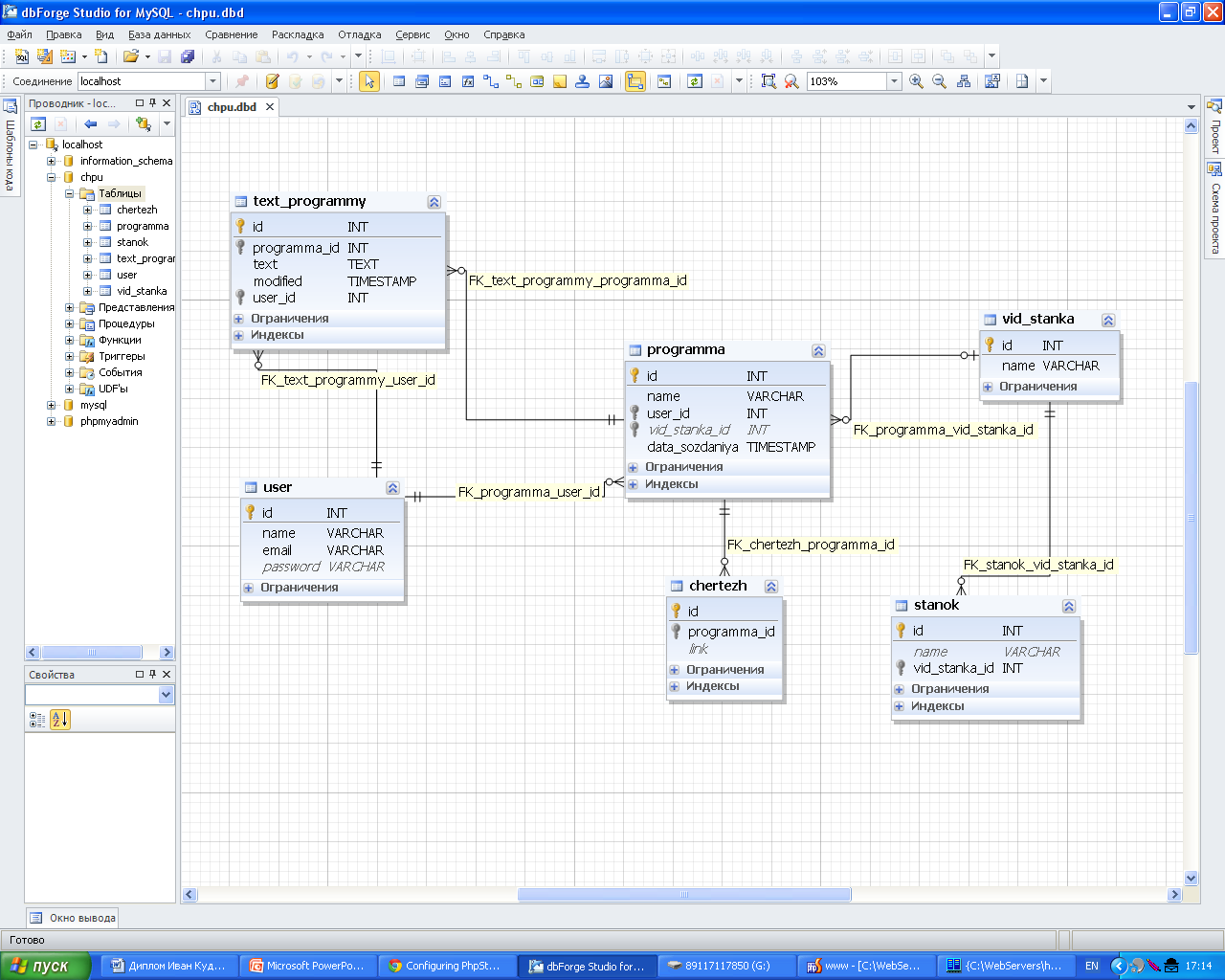
****

Рис. 6. Структура базы данных

**Таблица “user” –** хранение информации о пользователях системы. Нужна для идентификации и авторизации пользователей в системе.

Поля:

* **id –** идентификатор (номер) пользователя в системе;
* **name –** Фамилия имя отчество пользователя;
* **email –** электронная почта пользователя, используется как логин;
* **password –** md5-хеш сумма пароля пользователя.

MD5 – 128-битный алгоритм хеширования, разработанный профессором Рональдом Л. Ривестом из Массачусетского технологического института (Massachusetts Institute of Technology, MIT) в 1991 году. Предназначен для создания «отпечатков» или «дайджестов» сообщений произвольной длины и последующей проверки их подлинности.

Хранится вместо самого пароля на случай компрометации базы данных (т.е. когда какой-то кусок базы данных становится доступен злоумышленникам). Если пароль хранится в «прямом» виде, злоумышленники могут незаметно входить под любым пользователем системы.

**Таблица“programma” –** программа для станков с ЧПУ.

Поля:

* **id –** идентификатор (номер) программы в системе (для связей между таблицами);
* **name –** название программы;
* **user\_id–**автор программы;
* **vid\_stanka\_id –** для какого станка предназначена программа;
* **data\_sozdaniya–** дата создания программы, присваивается при .

Для одной программы хранится несколько текстов (версий) программы.

**Таблица “text\_programmy” –** версии текста программы.

Поля:

* **id –** идентификатор (номер) версии программы в системе;
* **programma\_id –** идентификатор программы, к которой относится данный текст (версия);
* **text –** сам текст программы (G-код) или разница (diff)с предыдущей версией программы;
* **modified –** дата и время последней модификации данного текста (данной версии). Для каждой версии своё поле modified.
* **comment –** комментарий к конкретной версии программы, можно хранить любые числа, строки и т.д.
* **user\_id –** пользователь, который создал данную версию программы.

**Таблица “vid\_stanka” –** виды станков.

Поля:

* **id –** идентификатор вида станка;
* **name–** название вида станка (например: Токарный, Зубодолбёжный, Зубофрезерный).

**Таблица “stanok” –** конкретный станок (с номером и т.д.).

Поля:

* **id –** идентификатор станка;
* **name –** идентифицирующее имя станка.
* **vid\_stanka\_id –** видстанка (ссылка на таблицу vid\_stanka).

**Таблица “chertezh” –** чертёж – картинка (JPEG, BMP и т.д.).

Поля:

* **id –** идентификатор станка;
* **programma\_id–** идентификатор программы.
* **link –** имя файла на сервере.
* **extension–** расширение файла.

**Функция Substr**(PHP 3, PHP 4, PHP 5) −Возвращает подстроку

Описание:

string**substr** ( string string, int start [, int length] )

**substr()** возвращает подстроку строки string длиной length, начинающегося с start символа по счету.

Если start неотрицателен, возвращаемая подстрока начинается в позиции start от начала строки, считая от нуля. Например, в строке 'abcdef', в позиции 0 находится символ 'a', в позиции 2 - символ 'c', и т.д.

**Пример 1.** Пример использования substr()

<?php

$rest = substr("abcdef", 1); *// возвращает "bcdef"*

$rest = substr("abcdef", 1, 3); *// возвращает "bcd"*

$rest = substr("abcdef", 0, 4); *// возвращает "abcd"*

$rest = substr("abcdef", 0, 8); *// возвращает "abcdef"*

*// к отдельным символам можно обращаться с помощью фигурных скобок*

$string = 'abcdef';

echo $string{0}; *// выводит a*

echo $string{3}; *// выводитd*

?>

Если start отрицательный, возвращаемая подстрока начинается с start символа с конца строки string.

**Пример 2.** Использование отрицательного start

<?php

$rest = substr("abcdef", -1);  *// возвращает "f"*

$rest = substr("abcdef", -2); *// возвращает "ef"*

$rest = substr("abcdef", -3, 1); *// возвращает "d"*

?>

Если length положительный, возвращаемая строка будет не длиннее length символов. Если длина строки string меньше или равна start символов, возвращается FALSE.

Если length отрицательный, то будет отброшено указанное этим аргументом число символов с конца строки string. Если при этом позиция начала подстроки, определяемая аргументом start, находится в отброшенной части строки, возвращается пустая строка.

**Пример 3.** Использование отрицательного length

<?php

$rest = substr("abcdef", 0, -1); *// возвращает "abcde"*

$rest = substr("abcdef", 2, -1); *// возвращает "cde"*

$rest = substr("abcdef", 4, -4); *// возвращает ""*

$rest = substr("abcdef", -3, -1); *// возвращает "de"*

?>

**Функция Stripslashes**(PHP 3, PHP 4, PHP 5)− Удаляет экранирование символов, произведенное функцией addslashes()

Описание:

string**stripslashes** ( stringstr )

Удаляетэкранирующиебэкслэши. (\' преобразуетсяв ', ит.д.). Двойные бэкслэши (\\) преобразуется в одиночные(\).

**stripslashes()** используется, например, когда директива конфигурации **magic\_quotes\_gpc** включена (она включена по умолчанию), и экранирование символов не требуется. Например, данные не вставляются в базу данных, а просто выводятся в браузер.

**Пример 1.** Пример использования **stripslashes()**

<?php

$str = "Is your name O\'reilly?";

echo stripslashes($str); *// выводит: Is your name O'reilly?*

?>

**Функцияstrip\_tags**(PHP 3 >= 3.0.8, PHP 4, PHP 5) −Удаляет HTML и PHP тэгиизстроки

Описание:

string**strip\_tags** ( string str [, string allowable\_tags] )

Эта функция возвращает строку str, из которой удалены HTML и PHP тэги. Для удаления тэгов используется автомат, аналогичный примененному в функции **fgetss()**.

Необязательный второй аргумент может быть использован для указания тэгов, которые не должны удаляться.

*Замечание:* Аргумент allowable\_tags был добавлен в PHP 3.0.13 и PHP 4.0b3. С версии PHP 4.3.0 удаляются также HTML комментарии.

Так как strip\_tags() не проверяет корректность HTML кода, незавершенные тэги могу привести к удалению текста, не входящего в тэги.

**Пример 1.** Пример использования **strip\_tags()**

<?php

$text = '

<p>Параграф.</p>

*<!-- Comment -->*

Ещенемноготекста';

echostrip\_tags($text);

echo "\n\n-------\n";

*// не удалять <p>*

echo strip\_tags($text, '<p>');

?>

*Этот пример выведет:*

Параграф.

Еще немного текста

-------

<p>Параграф.</p>

Еще немного текста

**Внимание:**

Эта функция не изменяет атрибуты тэгов, указанных в аргументе allowable\_tags, включая style и onmouseover.

С версии PHP 5.0.0 strip\_tags() безопасна для обработки данных в двоичной форме.

**Функцияstr\_replace**(PHP 3 >= 3.0.6, PHP 4, PHP 5) −Заменяет строку поиска на строку замены

Описание:

mixed**str\_replace** ( mixed search, mixed replace, mixed subject [, int &count] )

Эта функция возвращает строку или массив subject, в котором все вхождения search заменены на replace. Если не нужны сложные правила поиска/замены, использование этой функции предпочтительнее ereg\_replace() или preg\_replace().

С версии PHP 4.0.5, любой аргумент str\_replace() может быть массивом.

**Внимание:**

В версиях младше 4.3.3 эта функция содержит ошибку при одновременной передаче массивов в аргументах search и replace. Ошибка заключается в том, что пустые элементы массива search пропускались без перемещения к следующему элементу массива replace. Эта ошибка была исправлена в PHP 4.3.3. Если ваши скрипты использовали эту ошибку, то в них нужно удалить пустые элементы из массива search перед вызовом этой функции.

Если subject - массив, поиск и замена производится в каждом элементе этого массива, и возвращается также массив.

Если и search, и replace - массивы, то str\_replace() использует все значения массива search и соответствующие значения массива replace для поиска и замены в subject. Если в массиве replace меньше элементов, чем в search, в качестве строки замены для оставшихся значений будет использована пустая строка. Если search - массив, а replace - строка, то replace будет использована как строка замены для каждого элемента массива search.

**Пример 1.**  Примеры использования **str\_replace()**

<?php

*// присваивает<body text='black'>*

$bodytag = str\_replace("%body%", "black", "<body text='%body%'>");

*// присваивает: Hll Wrld f PHP*

$vowels = array("a", "e", "i", "o", "u", "A", "E", "I", "O", "U");

$onlyconsonants = str\_replace($vowels, "", "Hello World of PHP");

*// присваивает: You should eat pizza, beer, and ice cream every day*

$phrase = "You should eat fruits, vegetables, and fiber every day.";

$healthy = array("fruits", "vegetables", "fiber");

$yummy = array("pizza", "beer", "ice cream");

$newphrase = str\_replace($healthy, $yummy, $phrase);

*// начиная с версии 5.0.0 доступен аргумент count*

$str = str\_replace("ll", "", "good golly miss molly!", $count);

echo $count; // 2

?>

Замечание: Эта функция безопасна для обработки данных в двоичной форме.

Замечание: Начиная с PHP 5.0.0 количество произведенных замен может быть получено в необязательном аргументе count, который передается по ссылке. В версиях до PHP 5.0.0 этот аргумент недоступен.

**Функцияstrlen**(PHP 3, PHP 4, PHP 5) −Возвращает длину строки

Описание:

int**strlen** ( stringstring )

Возвращаетдлинустрокиstring.

**Пример 1.** Пример использования **strlen()**

<?php

$str = 'abcdef';

echo strlen($str); *// 6*

$str = ' ab cd ';

echostrlen($str); *// 7(в массиве 3 пробела, 4 символа)*

?>

**Функция quotemeta**(PHP 3, PHP 4)−закавычивает метасимволы.

Описание:

string**quotemeta** (stringstr)

Возвращает версию строки str с символами backslash (\) перед каждым из следующих символов:

. \\ + \* ? [ ^ ] ( $ )

**Пример 1.** Пример использования**quotemeta()**

<?php

$a = ". \\ + \* ?[ ^ ] ( $ )";

echo $a.'<br />';

echo quotemeta($a);

?>

Выведет:

. \ + \* ? [ ^ ] ( $ )

\. \\ \+ \\* \? \[ \^ \] \( \$ \)

**Функцияhtmlspecialchars**(PHP 3, PHP 4, PHP 5) − Преобразует специальные символы в HTML сущности

Описание:

string**htmlspecialchars** ( string string [, int quote\_style [, string charset]] )

В HTML некоторые символы имеют специальное значение и для сохранения своего значения должны быть преобразованы в HTML сущности. Эта функция возвращает строку, над которой проведены некоторые из таких преобразований. Этих преобразований достаточно для большинства задач веб-программирования. Если вам нужно преобразовать все возможные сущности, используйте **htmlentities()**.

Эта функция полезна при отображении данных, введенных пользователем, которые могут содержать нежелательные HTML тэги, например в форуме или гостевой книге. Необязательный второй аргумент quote\_style определяет режим обработки одиночных и двойных кавычек. В режиме по умолчанию, ENT\_COMPAT, преобразуются двойные кавычки, одиночные остаются без изменений. В режиме ENT\_QUOTES преобразуются и двойные, и одиночные кавычки, а в режиме ENT\_NOQUOTES и двойные, и одиночные кавычки остаются без изменений.

Производятся следующие преобразования:

'&' (амперсанд) преобразуется в '&amp;'

'"' (двойная кавычка) преобразуется в '&quot;' when ENT\_NOQUOTES is not set.

''' (одиночная кавычка) преобразуется в '&#039;' только в режиме ENT\_QUOTES.

'<' (знак "меньше чем") преобразуется в '&lt;'

'>' (знак "больше чем") преобразуется в '&gt;'

**Пример 1.**Примериспользования**htmlspecialchars()**

<?php

$new = htmlspecialchars("<a href='test'>Test</a>", ENT\_QUOTES);

echo $new; *// &lt;a href=&#039;test&#039;&gt;Test&lt;/a&gt;*

?>

Обратите внимание, что функция не производит других преобразований кроме описанных выше. Для преобразования всех HTML сущностей используйте htmlentities(). Поддержка необязательного второго аргумента была добавлена в PHP 3.0.17 и PHP 4.0.3.

Необязательный третий аргумент charset определяет кодировку, используемую при преобразовании. По умолчанию используется кодировка ISO-8859-1. Поддержка этого аргумента была добавлена в PHP 4.1.0.

**Функция settype**(PHP 3, PHP 4, PHP 5)−устанавливает типпеременной.

Описание:

bool**settype** (mixed var, string type)

Устанавливает тип type переменной var.

Возможные значения type:

"boolean" (или, начиная PHP 4.2.0, "bool")

"integer" (или, начиная PHP 4.2.0, "int")

"float" (только начиная с PHP 4.2.0; для более старых версий используйте не рекомендуемый вариант "double")

"string"

"array"

"object"

"null" (начинаяс PHP 4.2.0)

Возвращает TRUE при успехе; иначе возвращает FALSE.

**Пример 1. settype()**

$foo = "5bar";  *// string*

$bar = true; *// boolean*

settype($foo, "integer");  *// $foo is now 5 (integer)*

settype($bar, "string"); *// $bar is now "1" (string)*

**Функция session\_start**(PHP 4, PHP 5)−инициализирует данные сессии.

Описание:

bool**session\_start** (void)

session\_start() создаётсессию(илипродолжаеттекущуюнаосновеsessionid, переданногочерезGET-переменнуюиликуку).

Если вы хотите использовать именованную сессию, вы обязаны вызвать session\_name() до вызова session\_start().

Эта функция всегда возвращает TRUE.

**Примечание:** если вы используете сессии на базе кук, вы обязаны вызвать session\_start() перед каким-либо выводом в браузер.

session\_start() регистрирует внутренний обработчик вывода для URL, перезаписывая, если trans-sid включена. Если пользователь использует ob\_gzhandler или ob\_start(), порядок обработчика вывода важен для обеспечения правильного вывода. Например, пользователь обязан зарегистрировать ob\_gzhandler до старта сессии.

**Функция session\_destroy**(PHP 4, PHP 5)−разрушает все данные, зарегистрированные в сессии.

Описание:

bool**session\_destroy** (void)

session\_destroy() разрушает все данные, ассоциированные с текущей сессией. Не unset/разустанавливает никакие глобальные переменные, ассоциированные с сессией, и не unset куки сессии.

Эта функция возвращает TRUE при успехе и FALSE при неудаче разрушения данных сессии.

**Пример 1.** Разрушение сессии

<?php

*// Инициализировать сессию.*

*// Если you are using session\_name("something"), don't forget it now!*

session\_start();

*// Unset все переменные сессии.*

session\_unset();

*// Наконец, разрушить сессию.*

session\_destroy();

?>

**Функцияarray**(PHP 4, PHP 5) −Создатьмассив

Описание:

array**array** ([ mixed $... ] )

Возвратить массив параметров. Параметры могут быть заданы с индексом при помощи оператора =>.

**Замечание:**array() - это языковая конструкция для представления массивов, а не функция.

Синтакс "index =>values", разделённые запятыми, определяет индексы и их значения. Индекс может быть строкой или целым числом. Если индекс опущен, будет автоматически сгенерирован числовой индекс, начиная с 0. Если индекс - число, следующим сгенерированным индексом будет число, равное максимальному числовому индексу + 1. Обратите внимание, что если определены два одинаковых индекса, последующий переназначит предыдущий.

**Пример1.**Примериспользования**array()**

<?php

$fruits = array (

"fruits" => array("a" => "orange", "b" => "banana", "c" => "apple"),

"numbers" => array(1, 2, 3, 4, 5, 6),

"holes" => array("first", 5 => "second", "third")

);

?>

**Функция array\_merge**(PHP 4, PHP 5)− Слить два или большее количество массивов

Описание:

array**array\_merge** ( array $array1 , array $array2 [, array $... ] )

Функция array\_merge() сливает элементы двух или большего количества массивов таким образом, что значения одного массива присоединяются к значениям предыдущего. Результатом работы функции является новый массив.

Если входные массивы имеют одинаковые строковые ключи, тогда каждое значение, найденное позднее, будет заменять ранее найденное значение. Однако, если массивы имеют одинаковые числовые ключи, значение, упомянутое последним, не заменит исходное значение, а будет добавлено в конец массива.

**Пример 1.**Примериспользования**array\_merge()**

$array1 = array ("color" => "red", 2, 4);

$array2 = array ("a", "b", "color" => "green", "shape" => "trapezoid", 4);

$result = array\_merge ($array1, $array2);

print\_r($result);

Результат выполнения данного примера:

Array

(

[color] =>green

[0] => 2

[1] => 4

[2] => a

[3] => b

[shape] =>trapezoid

[4] => 4

)

**Функия implode**(PHP 3, PHP 4, PHP 5)−Объединяет элементы массива в строку

Описание:

string**implode** ( string glue, array pieces )

Возвращает строку, полученную объединением строковых представлений элементов массива pieces, со вставкой строки glue между соседними элементами.

**Пример 1.** Пример использования **implode()**

<?php

$array = array('lastname', 'email', 'phone');

$comma\_separated = implode(",", $array);

echo $comma\_separated; *// lastname,email,phone*

?>

**Замечание:** По историческим причинам, функции implode() можно передавать аргументы в любом порядке, однако для унификации с функцией explode() следует использовать документированный порядок аргументов.

**Замечание:** Начиная с версии 4.3.0 аргумент glue функции implode() является необязательным и по умолчанию равен пустой строке (''). Дляобеспечении обратной совместимости рекомендуется всегда передавать оба аргумента.

**Замечание:** Эта функция безопасна для обработки данных в двоичной форме.

**Функция shuffle**(PHP 3 >= 3.0.8, PHP 4, PHP 5)−Перемешать массив

Описание:

bool**shuffle** ( array&array )

Эта функция перемещает элементы массива в случайном порядке.

**Пример 1.** Пример использования **shuffle()**

<?php

$numbers = range(1, 20);

srand((float)microtime() \* 1000000);

shuffle($numbers);

while (list(, $number) = each($numbers)) {

echo "$number ";

}

?>

**Замечание:** Начиная с PHP 4.2.0, больше нет необходимости инициализировать генератор случайных чисел функциями srand() или mt\_srand(), поскольку теперь это происходит автоматически.

**Функция rand**(PHP 3, PHP 4, PHP 5)−Генерирует случайное число

Описание:

int**rand** ( [int min, int max] )

При вызове без параметров min и max, возвращает псевдослучайное целое в диапазоне от 0 до RAND\_MAX. Например, если вам нужно случайное число между 5 и 15 (включительно), вызовите rand (5, 15).

**Пример 1.**Пример**rand()**

<?php

echo rand() . "\n";

echo rand() . "\n";

echorand(5, 15);

?>

**Замечание:** На некоторых платформах (таких как Windows) RAND\_MAX всего лишь 32768. Чтобы расширить диапазон, используйте параметры min и max, или обратитесь к функции **mt\_rand()**.

**Замечание:** Начиная с PHP 4.2.0, больше нет необходимости инициализировать генератор случайных чисел функциями srand() или mt\_srand(), поскольку теперь это происходит автоматически.

**Конструкции возврата значений**

Конструкция **return**

Конструкция return возвращает значения, преимущественно из пользовательских функций, как параметры функционального запроса. При вызове return исполнение пользовательской функции прерывается, а конструкция return возвращает определенные значения.

Если конструкция return будет вызвана из глобальной области определения (вне пользовательских функций), то скрипт также завершит свою работу, а return также возвратит определенные значения.

Преимущественно, конструкция return используется для возврата значений пользовательскими функциями.

Возвращаемые значения могут быть любого типа, в том числе это могут быть списки и объекты. Возврат приводит к завершению выполнения функции и передаче управления обратно к той строке кода, в которой данная функция была вызвана.

**Пример** использования конструкции return для возврата значений типа integer:

<?php

function retfunct()

{

return 7;

}

echo retfunct();  *// выводит '7'.*

?>

**Функция imagecreatetruecolor** (PHP 4 >= 4.0.6)−создаётновоеизображение true color.

Описание:

resource **imagecreatetruecolor** (int x\_size, int y\_size)

imagecreatetruecolor() возвращает идентификатор чёрно-белого изображения размером x\_size на y\_size.

**Примечание:** Эта функция была введена в PHP 4.0.6 и требует GD 2.0.1 или новее.

**Функция imagecolorallocate**(PHP 3, PHP 4, PHP 5)−выделение цвета для изображения.

Описание:

int**imagecolorallocate** (resource image, int red, int green, int blue)

imagecolorallocate() возвращает идентификатор цвета, представляющий цвет, составленный из данных RGB-компонентов.Аргумент im возвращается от imagecreate().

red, green и blue это значения красного, зелёного и синего компонентов запрашиваемого цвета. Эти параметры являются целыми числами в диапазоне от 0 до 255. imagecolorallocate() обязана вызываться для создания каждого цвета, используемого в изображении image.

$white = imagecolorallocate ($im, 255, 255, 255);

$black = imagecolorallocate ($im, 0, 0, 0);

Возвращает -1, если выделение не удалось.

**Функция imagefilledrectangle**(PHP 3, PHP 4, PHP 5)−рисует заполненный прямоугольник.

Описание:

int**imagefilledrectangle** (resource image, int x1, int y1, int x2, int y2, int col)

imagefilledrectangle() создаёт заполненный прямоугольник цвета col на изображении image, начиная с верхней левой точки с координатами x1, y1 и заканчивая в правой нижней точке с координатами x2, y2.

0, 0 это верхний левый угол изображения.

**Функция imageline**(PHP 3, PHP 4, PHP 5)−рисует линию.

Описание:

int**imageline** (resource image, int x1, int y1, int x2, int y2, int col)

imageline() рисует линию от x1, y1 до x2, y2 (верхний левый угол изображения имеет координаты 0, 0) на изображении цветом col.

**Функция imagettftext**(PHP 3, PHP 4, PHP 5)−записывает текст на изображение с использованием шрифтов TrueType.

Описание:

array**imagettftext** (resource image, int size, int angle, int x, int y, int col, string fontfile, string text)

**imagettftext()**рисуетстроку text наизображении image, начинаяскоординат x, y (верхнийлевыйуголобозначается 0, 0), наугол angle цветом col, сиспользованиемфайлашрифта TrueType fontfile. В зависимости от используемой в РНР версии библиотеки GD, если fontfile не начинается с ведущего '/','.ttf' будет присоединено к имени файла, а библиотека будет пытаться искать файл с таким именем по всему пути шрифтов, определённому для библиотеки.

Координаты x, y будут определять базовую точку/basepoint первого символа (ориентировочно - левый верхний угол первого символа). Это отличается от imagestring(), где x, y определяют верхний правый угол первого символа.

*Angle* дан в градусах, где 0 градусов это чтение текста слева направо (направление 3 часа), а более высокие значения представляют ротацию против часовой стрелки (т.е. значение 90 должно дать чтение текста снизу вверх).

*Fontfile* это путь к используемому вами TrueType-шрифту.

*Text* это текстовая строка, которая может включать символьные последовательности UTF-8 (в форме &#123;) для доступа к символам за пределами первых 255.

*Col* это индекс цвета.Использование отрицательных значений даёт эффект отключения antialiasing.

**imagettftext()** возвращает массив из 8 элементов, представляющих 4 точки, образующих бокс текста. Порядок точек таков: нижняя левая, нижняя правая, верхняя правая, верхняя левая. Эти точки относительно текста, вне зависимости от угла, так что "верхний левый" означает верхний левый угол, когда вы смотрите на текст горизонтально.

Вот пример скрипта, производящего чёрный GIF 400x30 пикселов со словами "Testing..." белым шрифтом Arial.

**Пример 1.** Imagettftext

<?php

header ("Content-type: image/gif");

$im = imagecreate (400, 30);

$black = imagecolorallocate ($im, 0, 0, 0);

$white = imagecolorallocate ($im, 255, 255, 255);

imagettftext ($im, 20, 0, 10, 20, $white, "/path/arial.ttf", "Testing...Omega: &#937;");

imagegif ($im);

imagedestroy ($im);

?>

Примечание: Этафункциятребуетбиблиотек GD и FreeType.

**Функция imagejpeg**(PHP 3>= 3.0.16, PHP 4, PHP 5)−выводит изображение в браузер или файл.

Описание:

int**imagejpeg** (resource image [, string filename [, int quality]])

imagejpeg() создаёт JPEG-файл filename изизображения image. Аргумент image возвращёниз imagecreate().

Аргумент filename является необязательным и, если опущен, сырой поток изображения выводится напрямую.Для пропуска аргумента filename, с целью предоставления аргумента quality, просто используйте пустую строку (''). Отправляя image/jpeg content-type с помощью header(), вы можете создать PHP0-скрипт, который выводит JPEG-изображения напрямую.

**Примечание:** поддержка JPEG возможна только в том случае, если PHP был скомпилирован с GD-1.8 или новее.

quality является необязательным и имеет диапазон значений от 0 (наихудшее качество, наименьший файл) до 100 (наилучшее качество, наибольший файл). По умолчанию используется значение по умолчанию IJG quality (около 75).

Если вы хотите выводить Progressive JPEG, вам необходимо включить interlacing функцией imageinterlace().

**Функция header**(PHP 3, PHP 4, PHP 5)−отправляет "сырую" HTTP-шапку/header.

Описание:

int**header** (string string [, bool replace])

header() используется для отправки необработанных HTTP-шапок.

Необязательный параметр replace указывает, должна ли шапка замещать предыдущую аналогичную шапку, или нужно добавить вторую header того же самого типа. По умолчанию выполняется замещение, но, если вы передаёте FALSE в качестве второго аргумента, вы можете форсировать создание нескольких headers одного типа.

**Например:**

header('WWW-Authenticate: Negotiate');

header('WWW-Authenticate: NTLM',false);

Здесь два особых вызова шапок.В первом это header, начинающийся строкой "HTTP/" (регистр не имеет значения), используемый для создания HTTP статус-кода для отправки. Например, если вы сконфигурировали Apache для использования PHP-скрипта для обработки запросов отсутствующих файлов (используя директиву ErrorDocument), вам потребуется удостовериться, что ваш скрипт генерирует соответствующий статус-код.

<?php

header("HTTP/1.0 404 Not Found");

?>

**Примечание:** строка шапки HTTP-статуса всегда будет отправляться первой клиенту, независимо от того, идёт вызов header() первым или нет. Этот статус может быть переопределён вызовом header() с новой строкой статуса в любое время, если только HTTP headers уже не отправлены.

1. Описание параметров библиотеки (интерфейса подключения).

Для использования модуля нужно в вызывающемPHP-скрипте присвоить принятые от пользователя значения специальным переменным, которые будут обработаны внутри библиотеки. Далее следует поместить подключение библиотеки (при помощи include). После выполнения библиотеки параметры будут отфильтрованы нужным образом (в соответствии с их типом). Все не нужные символы и теги будут удалены.

**Интерфейс библиотеки:**

//Ведите сюда ваши переменные, для дальнейшей работы библиотеки. По умолчанию, все функции отключены, длявключение замените значения "vykl" на свои данные.

$namik = "vykl"; // переменная для проверки имени(логина) или пароля. Любого поля типа text

$famik = "vykl"; // переменная для проверки Фамилии или пароля. Любого поля типа text

$mailik = "vykl"; // переменная для проверки e-mail(адреса электронной почты).

$textik = "vykl"; // переменная для проверки общего текстового поля (textarea).

$textik1 = "vykl"; //переменная для проверки общего текстового поля (textarea). Преобразует специальные символы в HTML сущности, для безопасного их отображения на странице.

$teg = '<i><b>'; //перечислите теги, которые вы не хотите удалять из текстового поля.

$znakl = array("<",">","`","~","'","\"","\\","?","&","$","^","\*","{","}","[","]","(",")","%",";",":","/","!"); //массив вырезаемых из $namik символов

$znakt = array("`","~","'","\"","\\","$","^","\*","{","}","[","]"); //массив вырезаемых из $textik и $textik1 символов

$simn = 20; //кол-во символов вырезаемое из передаваемых данных в $namik, с начала.

$simm = 30; //кол-во символов вырезаемое из передаваемых данных в $mailik, с начала.

$simt = 300; //кол-во символов вырезаемое из передаваемых данных в $textik, с начала.

$provses = "vykl"; //переменная для для проверки переданного от пользователя значения капчи.Возвращает "2", если капча верна. "1" Если не верна.

$cifri = "vykl"; // переменная для для проверки цифрового поля int

$cifrf = "vykl"; // переменная для для проверки цифрового поля float

Если нужна проверка капчей, нужно в файле PHP передачи данных от пользователя вызвать файл капчи **sd\_captcha.php** в виде рисунка.

**Пример** вызова капчи:

<imgsrc="sd\_captcha.php" name="capt" />

Капча прорисует случайно сгенерированные символы, которые нужно будет ввести пользователю.

По умолчанию все функции библиотеки отключены (значением "vykl"). Для включения какой-либо функции проверки, просто подставьте свое значение вместо "vykl".

1. Реализация библиотеки

Наша библиотека состоит из:

1. Основного модуля библиотеки **protect.php**. В начале модуля есть переменные, которым нужно передать значения переменных поступивших от пользователя. Для его подключения нужно в используемом скрипте вызвать его **include ('protect.php');**
2. Модуля капчи **sd\_captcha.php**. Капча вызывается HTML тегом:

<img src="sd\_captcha.php" name="capt" >

1. Папки**shrift**содержащейшрифтыформата**\*.ttf**длякапчи.
2. Основной файл библиотеки protect.php

Листинг файла **protect.php**

<?php

// \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

// Модуль: protect.php Защита от SQL-атак, XSS-атак, и ботов(методом "капча")

// Дата создания: январь 2011 года

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//Ведите сюда ваши переменные, для дальнейшей работы библиотеки. По умолчанию, все функции отключены, длявключение замените значения "vykl" на свои данные.

$namik = "vykl"; // переменная для проверки имени(логина) или пароля. Любого поля типа text

$famik = "vykl"; // переменная для проверки Фамилии или пароля. Любого поля типа text

$mailik = "vykl"; // переменная для проверки e-mail(адреса электронной почты).

$textik = "vykl"; // переменная для проверки общего текстового поля (textarea).

$textik1 = "vykl"; //переменная для проверки общего текстового поля (textarea). Преобразует специальные символы в HTML сущности, для безопасного их отображения на странице.

$teg = '<i><b>'; //перечислите теги, которые вы не хотите удалять из текстового поля.

$znakl = array("<",">","`","~","'","\"","\\","?","&","$","^","\*","{","}","[","]","(",")","%",";",":","/","!"); //массив вырезаемых из $namik символов

$znakt = array("`","~","'","\"","\\","$","^","\*","{","}","[","]"); //массив вырезаемых из $textik и $textik1 символов

$simn = 20; //кол-во символов вырезаемое из передаваемых данных в $namik, с начала.

$simm = 30; //кол-во символов вырезаемое из передаваемых данных в $mailik, с начала.

$simt = 300; //кол-во символов вырезаемое из передаваемых данных в $textik и $textik1, с начала.

$provses = "vykl"; //переменная для для проверки переданного от пользователя значения капчи.Возвращает "2", если капча верна. "1" Если не верна.

$cifri = "vykl"; // переменная для для проверки цифрового поля int

$cifrf = "vykl"; // переменная для для проверки цифрового поля float

//\_\_\_\_\_\_\_\_\_\_\_начинаем работу!\_\_\_\_\_\_\_\_\_\_\_\_\_

if ($provses != "vykl"){

// стартуем сессию

session\_start ();

//Вырезаем нужное кол-во символов капчи

$provses = substr($provses,0,$\_SESSION['cap\_length\_max']);

//Проверяем совпадает ли переданный идентификатор с реальным

//Проверка капчи, функция возвращает true, если данные введны правильно. FALSE впротивномслучае.

function provsesiya ($provses){

if( $\_SESSION['captcha\_code'] == $provses){

//Разрушаемсессию

session\_destroy();

return true; //капчаверна

}

return false;

} }

//проверка на пустоту поля

if ($cifri != "vykl"){

settype($cifri, "integer"); // вернет только int значение переменной $cifri или FALSE(0);

}

//проверка на пустоту поля

if ($cifrf != "vykl"){

settype($cifrf, "float"); // вернет только float значение переменной$cifrf или FALSE(0);

}

//проверка на пустоту поля

if ($namik != "vykl"){

//обработака пользовательских данных:

$namik = str\_replace($znakl,"",strip\_tags(stripslashes(substr($namik,0,$simn))));

}

//проверка на пустоту поля

if ($famik != "vykl"){

//обработака пользовательских данных:

$famik = str\_replace($znakl,"",strip\_tags(stripslashes(substr($famik,0,$simn))));

}

//проверка на пустоту поля

if ($mailik != "vykl"){

//обработака пользовательских данных:

$mailik = strip\_tags(stripslashes(substr($mailik,0,$simm)));

// Проверка корректности адреса e-mail, функция возвращает true, если данные соответствуют регулярному выражению и кол-во символов больше 8.

function check\_email ($mailik)

{

if(preg\_match('|^[-0-9A-Za-z\_\.]+@[-0-9A-Za-z^\.]+\.[a-z]{2,6}$|i',$mailik) and strlen($mailik)>8) //регулярное выражение для проверки почты

{ return true; }

return false;

}

}

if($textik != "vykl") {

//обработакапользовательскихданных:

$textik = quotemeta(str\_replace($znakt, "",str\_replace/\*заменяемсимволывводана<br>\*/(array("\r\n","\r","\n"), "<br/>", strip\_tags(stripslashes(substr($textik,0,$simt)),$teg))));

}

if ($textik1 != "vykl"){

//обработака пользовательских данных:

$textik1 = quotemeta(str\_replace($znakt, "",str\_replace/\*заменяем символы ввода на <br>\*/(array("\r\n","\r","\n"), "<br/>", htmlspecialchars(strip\_tags(stripslashes(substr($textik1,0,$simt)),$teg)))));

}

//\_\_\_\_\_\_Проверка пользовательских данных закончена, можно приступать к работе с ними!\_\_\_\_

?>

После того, как в библиотеку введены все необходимые данные начинается ее работа. Переменная **$namik** будет обработана:

1. Будет вырезано нужное кол-во символов в зависимости от переменной $simn.
2. Будут удалены все бекслешы(\) функцией **stripslashes()**.
3. Будут вырезаны теги для защиты от XSS-атак функцией **strip\_tags()**.
4. Будут вырезаны все символы функцией **str\_replace()** переданные в массиве $znakl.

Переменная **$famik** будет обработана:

1. Будет вырезано нужное кол-во символов в зависимости от переменной $simn.
2. Будут удалены все бекслешы(\) функцией **stripslashes()**.
3. Будут вырезаны теги для защиты от XSS-атак функцией **strip\_tags()**.
4. Будут вырезаны все символы функцией **str\_replace()** переданные в массиве $znakl.

Переменные **$namik** и **$famik** предназначены для проверки полей типа text переданных от пользователя посредством формы. Эти переменные предназначены для проверок логина (имени), пароля, фамилии, и другой стоковой информации имеющей малый размер.Степень жесткости фильтрации можно регулировать массивом $znakl.

Переменная **$mailik** будет обработана:

1. Будет вырезано нужное кол-во символов в зависимости от переменной $simm.
2. Будут удалены все бекслешы(\) функцией **stripslashes()**.
3. Будут вырезаны теги для защиты от XSS-атак функцией **strip\_tags()**.
4. Проверяется кол-во символов. Их кол-во в почте должно всегда быть больше 8.
5. Проводится проверка переменной функцией **check\_email()**, проверяющей поступившие данные соответствию регулярному выражению:

**preg\_match**('|^[-0-9A-Za-z\_\.]+@[-0-9A-Za-z^\.]+\.[a-z]{2,6}$|i',$mailik)

Написанному специально для проверки почты. Функция возвращает **true,** если данные прошли проверку.**false**,если нет.

Переменная **$textik**будет обработана:

1. Будет вырезано нужное кол-во символов в зависимости от переменной $simt.
2. Будут удалены все бекслешы(\) функцией **stripslashes()**.
3. Будут вырезаны теги для защиты от XSS-атак функцией **strip\_tags()**, кроме тегов перечисленных в переменной **$teg**.
4. Все символы перевода строки ("\r\n","\r","\n") будут заменены натег<br />. Это сделано для удобства пользователей. Т.к. переменная**$textik** используется для передачи текстовых сообщений пользователем в гостевых книгах, форумах и др.
5. Будут вырезаны все символы функцией **str\_replace()** переданные в массиве $znakt.
6. Функцией **quotemeta()** будут прослешированны метасимволы:

. \\ + \* ? [ ^ ] ( $ )

Для защиты от SQL-атак.

Переменная **$textik1**будет обработана:

1. Будет вырезано нужное кол-во символов в зависимости от переменной $simt.
2. Будут удалены все бекслешы(\) функцией **stripslashes()**.
3. Будут вырезаны теги для защиты от XSS-атак функцией **strip\_tags()**, кроме тегов перечисленных в переменной **$teg**.
4. Функцией **htmlspecialchars()**специальные символы преобразуются вихhtml-сущности. Это делается для безопасного отображения тегов на странице.
5. Все символы перевода строки ("\r\n","\r","\n") будут заменены натег<br />. Это сделано для удобства пользователей. Т.к. переменная**$textik1**используется для передачи текстовых сообщений пользователем в гостевых книгах, форумах и др.
6. Будут вырезаны все символы функцией **str\_replace()** переданные в массиве $znakt.
7. Функцией **quotemeta()** будут прослешированны метасимволы:

. \\ + \* ? [ ^ ] ( $ )

Для защиты от SQL-атак.

Переменные **$textik**и **$textik1** предназначены для проверки полей типа <textarea> переданных от пользователя посредством формы. Эти переменные предназначены для проверки текстовых полей и др. текстовой информации имеющей большой размер.Степень жесткости фильтрации можно регулировать: массивом $znakt,кол-вом вводимых символов ($simt), разрешенными тегами (переменная $teg).

Переменные **$znakl** и**$znakt** **-** это массивы вырезаемых символов.

Переменные **$simn** и **$simm**и**$simt** – это кол-во вырезаемых символов.

Переменная **$provses**проверяет значение капчи переданное пользователем специально написанной для этого функцией **provsesiya()**со сгенерированным изображением капчи. Функция возвращает значение «true», если капча верна и «false», если нет. Если не нужна проверка, оставить значение **$provses**равным "vykl".

Переменная **$cifri** проверит соответствие значения типу **integer**. Вернет цифровое значение, если передано число. Если будет передана строка, вернет **FALSE(0)**.

Переменная **$cifrf** проверит соответствие значения типу **float**. Вернет цифровое значение, если передано число. Если будет передана строка, вернет **FALSE(0)**.

1. Подключаемый модуль библиотеки sd\_captcha.php

Этот PHP файл рисует капчу. Вначале файла присутствует меню настроек, где определяется:

1. ширина, высота капчи.
2. качество изображения
3. минимальное и максимальное кол-во символов используемых в капче.
4. Использование массивов с цифрами, латинскими буквами, русскими буквами.

Листинг файла**sd\_captcha.php**

<?php

session\_start();

$cap\_width=120;// ширинакапчи

$cap\_height=40;// высота капчи

$cap\_quality=10;// качество изображения, 100 = 100 процентов качество

$cap\_length\_min=4;// минимальное кол-во символов в коде

$cap\_length\_max=5;// максимальное кол-во символов в коде

$cap\_digital=1;// true (1) - в коде разрешено использовать цифры

$cap\_latin\_char=1;// true (1) - в коде разрешено использовать латинские символы

$cap\_russian\_char=0;// true (1) - в коде разрешено использовать русские символы

// \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

function code\_generic($\_length,$\_digital=1,$\_latin\_char=1,$\_russian\_char=0)

// генерацияпароля

// \_legnth - длина пароля

// \_digital - 1 разрешена генерация цифр в пароле

// \_latin\_char - 1 разрешена генерация латинских букв в пароле

// \_russian\_char - 1 разрешена генерация русский букв в пароле

{

$dig=array(0,1,2,3,4,5,6,7,8,9);// массив с цифрами

// массив с русскими буквами

$rus=array('а','б','в','г','д','е','ж','з','и','к','л','м','н','о',

'п','р','с','т','у','ф','х','ц','ч','ш','щ','ь','ы','ъ','э','ю','я','ё','й');

// массив с лотинскими буквами

$lat=array('a','b','c','d','e','f','g','h','j','k','l','m','n','o',

'p','q','r','s','t','u','v','w','x','y','z');

$main=array();

// если требуется объединяем массив из цифр

if ($\_digital) $main=array\_merge($main,$dig);

// если требуется объедениям массив из латинских буковок

if ($\_latin\_char) $main=array\_merge($main,$lat);

// если требуется объеденяем массив из русских буковок

if ($\_russian\_char) $main=array\_merge($main,$rus);

shuffle($main);// перемешать массив в случайном порядке

// вырезаем нужное количество символов из перемешенного массива, и аля паролька готова ;)

$pass=substr(implode('',$main),0,$\_length);

return $pass;// возвращаемрезультат

}

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

$l=rand($cap\_length\_min,$cap\_length\_max);// определяемкол-восимволоввкоде

$code=code\_generic($l,$cap\_digital,$cap\_latin\_char,$cap\_russian\_char);// генерируемкод

$\_SESSION['captcha\_code']=$code;// запоминаем код для будущей сверки

$\_SESSION['cap\_length\_max']=$cap\_length\_max;//запоминаем макс. кол-во символов капчи

$canvas=imagecreatetruecolor($cap\_width,$cap\_height);// создаем прямоугольник куда и будем рисовать

$c=imagecolorallocate($canvas,rand(230,250),250,250);// цветфоназаполнения

imagefilledrectangle($canvas,0,0,$cap\_width,$cap\_height,$c);// заливаемкартинкуфоновымцветом

$count=strlen($code);// узнаемдлинукода

$color\_text=imagecolorallocate($canvas,rand(20,150),rand(20,150),190); //цветтекста

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_массивсошрифтами\_\_\_\_\_\_\_\_\_\_\_\_\_\_

$shrift[0] = "shrift/a\_CampusGrav.ttf" ;

$shrift[1] = "shrift/download\_fonts\_ttf-rus\_CampusOtl\_Bold.ttf" ;

$shrift[2] = "shrift/download\_fonts\_ttf-rus\_Bookman\_Old\_Style.ttf" ;

$shrift[3] = "shrift/download\_fonts\_ttf-rus\_BoisterBlack.ttf" ;

$shrift[4] = "shrift/download\_fonts\_ttf-rus\_TrooverRoman.ttf" ;

$shrift[5] = "shrift/ELEPHNT.ttf" ;

$shrift[6] = "shrift/IMPRISHA.ttf" ;

//выбор шрифта

$sh = rand(0,6);

//создаём шум на фоне в виде коротеньких линий

for ($i=0; $i<=80; $i++) {

$xn = rand(2,$cap\_width);

$yn = rand(2,$cap\_height);

$xk = $xn + rand(-10,10);

$yk = $yn + rand(-10,10);

imageline($canvas,$xn,$yn,$xk,$yk,$color\_text);

}

for($it=0;$it<$count;$it++)// прорисуемкаждуюбуковку

{ $letter=$code[$it];// берембуковку

//imagestring($canvas,6,(10\*$it+20),$cap\_height/4,$letter,$color\_text);

imagettftext($canvas,20,rand(-35,35),(18\*$it+rand(10,12)),$cap\_height/1.5,$color\_text,$shrift[$sh],$letter);

}

// создаем соответствующий заголовок для корректности вывода изображения

// делаем все так, чтобы браузеры не кэшировали нашу картинку

header('Expires: Mon, 26 Jul 1997 05:00:00 GMT');

header('Cache-Control: no-store, no-cache, must-revalidate');

header('Cache-Control: post-check=0, pre-check=0',false);

header('Pragma: no-cache');

header('Content-Type: image/jpeg');

imagejpeg($canvas,null,$cap\_quality);// непосредственно создаем картинку

?>

После определения начальных значений капча определяет:

1. нужное кол-во символов.
2. Затем начинается генерация кода специально написанной для этого функцией **code\_generic()**.
3. Сгенерированный код запоминается в переменную $\_SESSION['captcha\_code'] для будущей сверки.
4. Также запоминается длина пароля в переменную $\_SESSION['cap\_length\_max'].
5. Создается прямоугольник, куда будет нарисована капча.
6. Создается цвет фона для прямоугольника.
7. Прямоугольник заливается созданным фоновым цветом.
8. Узнаем длину кода для будущей прорисовки символов.
9. Создается цвет символов кода.
10. Случайным образом выбирается шрифт для кода, из массива шрифтов.
11. Создается «шум» на картинке, в виде коротеньких линий.
12. Прорисовывается каждый символ нашего кода.
13. Делаем все так, чтобы браузеры не кэшировали нашу картинку.
14. Готовое изображение с нашим кодом выдается в браузер.

Работа модуля **sd\_captcha.php**завершена. Вызов картинки с кодом осуществляется простой вставкой HTML тега img:

<img src="sd\_captcha.php" name="capt" >

Имя картинки может быть любым. При каждом вызове картинки, код на ней будет обновляться.

1. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ СИСТЕМЫ
2. Вход в систему, авторизация и аутентификация

Скриншоты.

1. Использование для защиты от автоматической отправки сообщений, XSS-атак. Примеры фильтрации параметров SQL-запросов.

Для проверки работоспособности разработанной библиотеки мной специально был написан локальный сайт **proba.ru**. Его работа состоит в получении от пользователя имени, фамилии, пожелания и любимого числа, для записи их в базу данных MYSQL. И в дальнейшем выводе в браузер этой информации из базы данных с точным временем и датой получения этих данных.

Сайт состоит из файлов:

1. **index.php** – главная страница сайта, выводит значения из БД и содержит форму ввода новой информации от пользователей.
2. **addrecord.php** – скрипт добавления поступивших данных в БД. Подключает нашу библиотеку для фильтрации данных.
3. **БД MySQL** – хранится на локальном сервере. База называется 'progtest', содержит одну таблицу `proba`.
4. Файлы нашей библиотеки:
   1. **protect.php** – основной модуль нашей библиотеки.
   2. **sd\_captcha.php** – модуль капчи.
   3. **shrift** – папка содержащая шрифты **\*.ttf**для массива в **sd\_captcha.php**.

Листинг**addrecord.php**

<?php

include('protect.php');

if (provsesiya($provses)){

if ($namik== "" or $famik== "" or $textik == "" or $cifri == ""){

echo '<div align=center>';

echo "вы что то не ввели!";

echo "<a href=\"index.php\">наглавную</a>";

echo '</div>';

} else{

$vrm = time();

//запись в базу данных

//конекткбазе

if ($con = mysql\_connect('localhost','root',''))

{echo "есть контакт!";}

else echo "неудача! Проверь настройки!";

//выбор базы

if(mysql\_select\_db('progtest', $con))

// записьвтаблицу

{ mysql\_query("INSERT INTO `proba` (`vr`,`imya`,`fam`,`text`,`chis`) VALUES ('$vrm','$namik','$famik','$textik','$cifri')");

echo "<p align=center>запись добавлена! <br/>";

echo "<a href=\"index.php\">наглавную</a></p>";

echo $textik;

}

else echo "her";

}

} else echo "<a href=\"index.php\">назад, кодневерен</a>";

?>

Модулю библиотеки **protect.php**были переданы нужные переменные для проверки. Это делается единожды при настройке библиотеки:

$namik = $\_POST['imya']; // переменная для проверки имени(логина) или пароля. Любого поля типа text

$famik = $\_POST['fam']; // переменная для проверки Фамилии или пароля. Любого поля типа text

$textik = $\_POST['text']; // переменная для проверки общего текстового поля (textarea).

$provses = $\_POST['code2']; //переменная для проверки переданного от пользователя значения капчи.Возвращает "2", если капча верна. "1" Если не верна.

$cifri = $\_POST['chis']; // переменная для для проверки цифрового поля int.

После инклюда библиотеки данные проверены и с ними можно работать.

1. Тестирование капчи

Созданная мной капча была протестирована на распознавание человеком и популярной программой для распознавания текста «FineReader 9.0 Professional Edition». Результаты тестирования приведены в гистограмме «Статистика распознавания капчи».



Для распознавания были использованы 10 случайно сгенерированных выводов капчи:

C:\Users\леночка\Desktop\capt\sd_captcha9.jpegC:\Users\леночка\Desktop\capt\sd_captcha8.jpegC:\Users\леночка\Desktop\capt\sd_captcha7.jpegC:\Users\леночка\Desktop\capt\sd_captcha6.jpegC:\Users\леночка\Desktop\capt\sd_captcha5.jpegC:\Users\леночка\Desktop\capt\sd_captcha4.jpegC:\Users\леночка\Desktop\capt\sd_captcha3.jpegC:\Users\леночка\Desktop\capt\sd_captcha2.jpegC:\Users\леночка\Desktop\capt\sd_captcha1.jpegC:\Users\леночка\Desktop\capt\sd_captcha.jpeg

Из приведенной статистики видно, что программа «FineReader 9.0» ни разу не распознала капчу. Человек распознал 6 из 10 вариантов. Из этого следует, что полученная капча достаточно надежна для использования. И для упрощения распознавания капчи человеком, совсем не трудно поставить рядом с капчей кнопку «обновить», которая будет заново вызывать скрипт прорисовки новой капчи.

1. ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ
2. Концепция технико-экономического обоснования

Значение информационной безопасности в коммерческих структурах всегда было велико, однако долгое время недооценивалось руководителями предприятий. На сегодняшний день существует тенденция увеличения значимости вопросов информационной безопасности в глазах руководства компаний. В последние 7 лет в России наблюдается экспоненциальный рост числа компаний, предлагающих услуги, связанные с обеспечением информационной безопасности. Однако квалифицированных специалистов не становится больше, из-за создающегося дефицита уровень зарплат специалистов по ИБ растет, одновременно растет и стоимость услуг компаний, нанимающих этих специалистов.

Стоит отметить, что для небольших компаний, имеющих в штате одного системного администратора, совершенно нерентабельно нанимать дополнительно специалиста по ИБ, к тому же и высокооплачиваемого. Поэтому у руководителей таких компаний не остается другого выбора, как прибегать к помощи компаний, предоставляющих услуги ИБ.

Разрабатываемый в данном дипломном проекте научно-технический продукт представляет собой библиотеку, которая обеспечивает комплексную защиту сайта на языке программирования PHP. Она включает в себя защиту от SQL-атак, XSS-атак, защиту от ботов методом капча, PHP-инклюдинга и др.

1. Потребительские свойства

Разработанный программный продукт, встраиваемая библиотека методом **include()**, фильтрует получаемые данные от пользователя.Что позволяет использовать ее как уже в готовых сайтах для повышения их безопасности, так и на стадии разработки нового сайта. Что делает ее уникальным и востребованным продуктом.

Библиотека работает через интерфейс веб-браузера, поэтому не требует вложений в специализированное платное ПО. Единственными ресурсами, необходимыми для использования библиотеки, является компьютер с любой современной операционной системой, подключенный к интернету.При этом система учитывает, что для ее использования не нужно иметь навыков специалиста по информационной безопасности. Все взаимодействие с пользователем системы ведется через интуитивно понятный интерфейс.

1. Затраты на разработку

Расчет себестоимости выполненных работ осуществляется по следующим статьям:

* расходы на оплату труда (относится заработная плата научных сотрудников, инженеров и прочего инженерно-технического персонала, непосредственно занятых выполнением конкретной работы);
* отчисления на социальные нужды (относят затраты, связанные с выплатой единого социального налога);
* накладные расходы (включаются расходы на управление и хозяйственное обслуживание);
* материалы (относятся затраты на сырье, основные и вспомогательные материалы, покупные полуфабрикаты и комплектующие изделия, необходимые для выполнения конкретной работы) [13].

**Расходы на оплату труда**

В работе непосредственно были задействованы только руководитель и исполнитель. Поэтому расходы на выплату заработной платы будут учитываться только в отношении данных лиц.

Сроки проведения работы – 4 месяцев. Ставка руководителя – 20 000 рублей в месяц. Ставка исполнителя – 10 000 рублей в месяц. Представим расчеты в виде таблицы 4.1.

Таблица 4.1 - Расходы на оплату труда.

|  |  |  |
| --- | --- | --- |
| Работник | Величина з/п., руб. | Выплаты по з/п за 4 месяца, руб. |
| Руководитель | 20 000 | 80 000 |
| Исполнитель | 10 000 | 40 000 |
| **Итого** | **30 000** | **120 000** |

Дополнительная заработная плата составляет 12% от суммы основной заработной платы:

С дз = 120 000 \*0,12=14 440 рублей.

**Отчисления на социальные нужды**

Рассчитаем величину отчислений на социальные нужды за четыре месяца.

С 1 января 2010 года работодатели вместо ЕСН будут платить страховые взносы в государственные внебюджетные фонды. Страховые взносы на обязательное социальное страхование от несчастных случаев на производстве и профессиональных заболеваний сохраняются. Порядок их расчетов установлен Федеральным законом от 24.07.1998 № 125-ФЗ.

Тарифы взносов во внебюджетные фонды для работодателей, применяющих общую систему налогообложения, в 2010 году останутся в размерах действующих в 2009 году ставок ЕСН (ст. 57 Закона № 212-ФЗ). Отметим, что "старые" ставки на 2010 год оставлены исключительно ввиду кризиса.

Законодатели решили увеличить ставки только с 2011 года (ст. 12 Закона № 212-ФЗ). Для наглядности увеличение и разбивку новых ставок по фондам представим в виде таблицы:

|  |  |  |
| --- | --- | --- |
| **Наименование фонда** | **Ставка взносов в % при общей системе налогообложения** | |
| **2009-2010 гг.** | **2011 г.** |
| ПФР | 20,0\* | 26,0\*\* |
| ФСС | 2,9 | 2,9 |
| ФФОМС | 1,1 | 2,1 |
| ТФОМС | 2,0 | 3,0 |
| **Итого** | **26,0** | **34,0** |

Расчет производится по формуле



где  - величина заработной платы;  - величина дополнительной заработной платы; - Ставка взносов, %. На момент проведения работы ставка единого социального налога равна 26%. Исходя из имеющихся данных, вычислим размер отчислений:

С сн =(120 000 + 14 440)\*0,26 = 34 954,4 рублей.

**Расходы на материалы**

Стоимость расходных материалов, необходимых для выполнения разработки, определяется исходя из величины их расхода, действующих цен. Расчет расходов по статье "Материалы" представлен в таблице 4.2.

Таблица 4.2 - Расходы на материалы.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование | Единица измерения | Количество | Цена, руб. | Сумма, руб. |
| Бумага для принтера | пачка в 500 листов | 5 | 100 | 500,00 |
| Чернила для принтера | Чернильниц, штук | 2 | 1 500 | 3 000,00 |
| Книга «Д. Н. Колисниченко - Самоучитель PHP 5» | штук | 1 | 430 | 430,00 |
| Транспортно-заготовительные расходы (10% от материалов) | | | | 393,00 |
| **Итого** | | | | **4 233,00** |

**Накладные расходы**

Накладные расходы составляют 20% от всех расходов [13].

С нр =159 187,4\*0,2 = 31 837,48 рублей.

**Калькуляция себестоимости разработки**

Итоговая калькуляция себестоимости приведена в таблице 4.3.

Таблица 4.3 - Калькуляция себестоимости продукции.

|  |  |
| --- | --- |
| **Статья расходов** | **Сумма, руб.** |
| Материалы | 4 233,00 |
| Основная заработная плата | 120 000,00 |
| Дополнительная заработная плата | 14 440 |
| Отчисления на социальные нужды | 34 954,4 |
| Накладные расходы | 31 837,48 |
| **ИТОГО** | **205 464,88** |

В результате произведенных вычислений получаем себестоимость разработки 205 464,88рублей.

1. Трудоемкость и календарный план

Календарный план разработки программного продукта можно представить в виде следующей последовательности:

1. Сбор научно-технической документации изучаемой области, изучение существующих методов.
2. Программная реализация основных методов на языке программирования PHP и анализ результатов их работы с помощью существующих показателей.
3. Разработка программного продукта на основании проведенного анализа.
4. Написание пояснительной записки к дипломному проекту.

Календарный план работ, включающий все вышеописанные этапы сведен в таблице 4.4.

Таблица 4.4 Ленточный график проведения разработки программного продукта.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| № | Этапы работы | | | | | | | | | Исполнитель | | | Длительность, недели (5 дней) | | | |
| 1 | Руководство проекта, написание технического задания, контроль выполнения этапов проекта и оценка качества полученных результатов. | | | | | | | | | Руководитель | | | 16 | | | |
| 2 | Сбор научно-технической документации изучаемой области, изучение существующих методов. | | | | | | | | | Специалист | | | 2 | | | |
| 3 | Программная реализация основных методов на языке программирования PHP и анализ результатов их работы с помощью существующих показателей | | | | | | | | | Специалист | | | 4 | | | |
| 4 | Разработка программного продукта на основании проведенного анализа | | | | | | | | | Специалист | | | 8 | | | |
| 5 | Написание пояснительной записки к дипломному проекту | | | | | | | | | Специалист | | | 2 | | | |
| № | Продолжительность работ, недели (5-дневная рабочая неделя) | | | | | | | | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| 1 |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |

1. Комплексная оценка эффективности

Научно-техническим эффектом разработки программного продукта, основанного фильтрации входящих пользовательских данных, является возможность контролировать безопасность этих данных. В первую очередь следует отметить то, что аналогов у данной системы на данный момент на рынке нет, либо же они абсолютно неизвестны. В случае успеха это обеспечит стремительный рост популярности научно-технического продукта.

Также необходимо учесть, что, так как аналоги на данный момент отсутствуют, то существует риск, что продукт после выхода на рынок может оказаться невостребованным, если пользователи не увидят для себя выгоды его использования.

В общем можно заключить, что для успешной коммерческой реализации данного научно-технического продукта необходимо:

* наличие хорошо спроектированного интерфейса пользователя;
* разработка и проведение продуманной маркетинговой политики для привлечения пользователей и поставщиков.

1. Выводы

На основании анализа предполагаемых потребительских свойств разрабатываемой системы, расчетов себестоимости разрабатываемой системы и существующего положения дел в области информационной безопасности сайтов можно сделать вывод, что экономическая эффективность разработки и применения продукта не доказана. Разработка продукта требует значительных финансовых вложений и при этом является рискованным предприятием.

1. ВОПРОСЫ ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ
2. Общие положения

Под **интеллектуальной собственностью** понимают особый вид гражданских прав (исключительное право) в отношении результатов интеллектуальной деятельности, таких как изобретения, промышленные образцы (дизайн), компьютерные программы, другие произведения науки, произведения литературы, искусства, которые принято называть **объектами интеллектуальной собственности,** а также различных средств индивидуализации производителя товаров и услуг, таких как товарные знаки, знаки обслуживания, фирменные наименования и др. Основным содержанием таких прав является монополия их владельца на использование этих объектов, включая право запретить или разрешить их использование другим, а также право переуступить другому лицу эти правомочия или отказаться от них вовсе.

Согласно определению интеллектуальной собственности, принятому в российском законодательстве, а также на основании определения Стокгольмской конференции от 14 июля 1967 г., программы для ЭВМ (компьютерные программы) и базы данных относятся к объектам интеллектуальной собственности. Программам для ЭВМ и базам данных предоставляется охрана нормами авторского права как литературным произведениям в соответствии с Бернской конвенцией, причем программы для ЭВМ охраняются как литературные произведения, а базы данных - как сборники.

В Российской Федерации вопросы предоставления правовой охраны программам для ЭВМ и базам данных регулируются Гражданским кодексом РФ, Часть 4 (ГК РФ Ч.4).

### Авторское право на программу для ЭВМ и базу данных

Предпосылкой охраноспособности программы для ЭВМ и базы данных является их творческий характер, т. е. они должны быть продуктом личного творчества автора. Творческий характер деятельности автора предполагается до тех пор, пока не доказано обратное.

Авторское право на программы для ЭВМ или базы данных возникает в силу их создания. Для признания и осуществления авторского права на программу для ЭВМ и базу данных не требуется депонирования, регистрации или соблюдения иных формальностей. Авторское право распространяется на любые программы для ЭВМ и базы данных как выпущенные, так и не выпущенные в свет, представленные в объективной форме, независимо от их материального носителя.

Таким образом, только сам факт создания программы или базы данных, зафиксированных в объективной форме, является основанием возникновения авторского права на эти объекты. С этого момента права автора или иного правообладателя защищаются законом.

Права в отношении программ для ЭВМ и баз данных подразделяются на личные неимущественные и исключительные права.

Личные права включают право авторства, право на имя и право на неприкосновенность (целостность), право на обнародование программы для ЭВМ или базы данных. Они связаны непосредственно с автором программы для ЭВМ или базы данных: принадлежат лицу, чьим творческим трудом созданы программа для ЭВМ или база данных - автору, являются неотчуждаемыми, т. е. не могут быть переуступлены другому лицу, и не ограничены каким-либо сроком. Авторские права на данный программный продукт принадлежат мне, автору, Шустрову Александру Борисовичу, т.к. создан моим творческим трудом.

Исключительные права связаны с правомочием правообладателя осуществлять и/или разрешать осуществление определенных действий с программами для ЭВМ или базами данных, а именно: выпуск в свет (опубликование); воспроизведение (полное или частичное) в любой форме, любыми способами; распространение; модификацию и иное использование. Они могут принадлежать автору или иному правообладателю (физическому или юридическому лицу), могут быть переуступлены другому лицу (отчуждены) и срок их действия ограничен.

Исключительные (имущественные) права непосредственно связаны с понятием «использования» программ для ЭВМ и баз данных, которое определено следующим образом: «Использование программы для ЭВМ или базы данных - это выпуск в свет, воспроизведение, распространение и иные действия по их введению в хозяйственный оборот (в том числе в модифицированной форме)». Каждая составляющая понятия использования программы для ЭВМ или базы данных имеет конкретное содержание, которое также определено законом:

1. **выпуск в свет (опубликование) -** *"...это предоставление экземпляров программы для ЭВМ или базы данных с согласия автора неопределенному кругу лиц (в том числе путем записи в память ЭВМ и выпуска печатного текста) при условии, что количество таких экземпляров должно удовлетворять потребности этого круга лиц, принимая во внимание характер указанных произведений"*;
2. **воспроизведение** - *"...это изготовление одного или более экземпляров программы для ЭВМ или базы данных в любой материальной форме, а также их запись в память ЭВМ"*;
3. **распространение** - *"...это предоставление доступа к воспроизведенной в любой материальной форме программе для ЭВМ или базе данных, в том числе сетевыми и иными способами, а также путем продажи, проката, сдачи внаем, предоставления взаймы, включая импорт для любой из этих целей"*.

В целях **оповещения о своих правах** правообладатель *"... может, начиная с первого выпуска в свет программы для ЭВМ или базы данных, использовать знак охраны авторского права, состоящий из трех элементов: буквы С в окружности или в круглых скобках; наименования (имени) правообладателя; года первого выпуска программы для ЭВМ или базы данных в свет"*. Знак охраны авторского права может проставляться на упаковке, на самом программном продукте, а также на всех сопроводительных и дополнительных материалах, однако это не является обязательным. Следует иметь в виду, что сам знак ничего не защищает (защищает закон!) и что наличие или отсутствие знака охраны никак не связано с возникновением авторского права. Знак охраны авторского права - это цивилизованная форма предупреждения желающих использовать данный объект, что права на него охраняются законом, с одновременным сообщением о том, кому эти права принадлежат.

Исключительные (имущественные) права на программу или базу данных переходят по наследству в установленном законом порядке, и их можно реализовать в течение срока действия авторского права. Права на программу для ЭВМ или базу данных не связаны с **правом собственности на материальный носитель**, на котором они зафиксированы. Передача прав на материальный носитель не влечет за собой передачи каких-либо прав на программу для ЭВМ или базу данных. Иными словами, передача носителя информации (например, дискеты) с зафиксированной на нем программой третьему лицу не означает передачи каких-либо прав на эту программу.

### Правообладание

Если человек (или группа людей) самостоятельно, по личной инициативе создал программу для ЭВМ или базу данных, то он является одновременно и автором, и правообладателем созданного произведения, что позволяет ему по собственному усмотрению использовать эту программу (или базу данных) в личных целях, продавать, раздавать бесплатно, разрешать тиражировать и распространять или иным образом осуществлять свои имущественные права.

Если программа для ЭВМ или база данных разрабатываются по заданию (заказу), то заказчик заключает с разработчиком договор на создание данного объекта, который называется авторским договором заказа. По авторскому договору заказа автор обязуется создать произведение в соответствии с условиями договора и передать его, а также имущественные права на него заказчику, а заказчик обязуется в счет обусловленного договором вознаграждения выплатить автору аванс.

Совсем иначе обстоит дело, если автор создал программу или базу данных в порядке выполнения своих служебных обязанностей или по отдельному заданию работодателя, т. е. предприятия, с которым он связан трудовыми отношениями. В этом случае по закону все имущественные права на созданные программы для ЭВМ и базы данных принадлежат работодателю, если договором между ним и автором не предусмотрено иное.

Т.к. созданный программный продукт был разработан в рамках дипломного проекта, по собственной инициативе автора, то правообладателем является автор.

### Передача исключительных прав на программу для ЭВМ и базу данных

Исключительные (имущественные) права на программу для ЭВМ и базу данных могут быть переданы полностью (полная уступка прав) или частично (передача прав на использование) другим физическим или юридическим лицам. Передача исключительных прав должна быть оформлена заключением договора в письменной форме, который называется «Договор о полной уступке исключительных прав».

При полной уступке исключительных прав автором или иным правообладателем правоприобретатель (физическое или юридическое лицо) получает возможность осуществлять все эти права в полном объеме: использовать программу для ЭВМ или базу данных самостоятельно, например, путем изготовления и распространения экземпляров программ или баз данных, разрешать и запрещать третьим лицам их использование, уступить все права или отказаться от них вовсе, - в то время как прежний правообладатель лишается этой возможности.

Права на использование программы для ЭВМ или базы данных могут предоставляться на основании договора с правообладателем, который может быть двух видов: исключительная лицензия и неисключительная (простая) лицензия. В договоре должны быть определены следующие основные условия: объем и способы использования, порядок выплаты вознаграждения и срок действия договора, а также территория, на которой используется данная программа для ЭВМ или база данных. В случае исключительной лицензии передаются права на монопольное использование объекта лицензии одному лицу в объеме и на срок, определяемые условиями договора. В случае неисключительной лицензии передаются права на использование объекта лицензии в пределах, обусловленных договором, при этом правообладатель оставляет такие же права за собой и может выдавать неисключительные лицензии на тот же объект на тех же или иных условиях третьим лицам.

Следует отметить, что для договоров, предусматривающих передачу исключительных прав, законодатель использует также термин «авторский договор», заключаемый в отношении объектов авторского права, к которым относятся и программы для ЭВМ, и базы данных. Термин «лицензия» следует из сложившихся обычаев делового оборота.

### Нарушение прав на программу для ЭВМ и базу данных

Специфика программ для ЭВМ и баз данных такова, что они очень уязвимы в смысле их незаконного использования (прежде всего, путем копирования и распространения копий). Незаконно изготовленные (скопированные) или используемые экземпляры программы для ЭВМ или базы данных называются контрафактными, а несанкционированное использование чужих программ или баз данных путем выпуска в свет, воспроизведения (полного или частичного), распространения, модификации (в том числе перевода с одного языка на другой), иного использования считается нарушением имущественных прав на программы для ЭВМ или базы данных, т. е. нарушением авторского права.

1. Комплект основных документов для официальной регистрации программы для ЭВМ

В ст. 1262 ГК РФ Ч.4 закреплено право автора или иного правообладателя на государственную регистрацию программы для ЭВМ или базы данных: *"Правообладатель в течение срока действия исключительного права на программу для ЭВМ или базу данных может по своему желанию зарегистрировать такую программу для ЭВМ или такую базу данных в федеральном органе исполнительной власти по интеллектуальной собственности "*. Исключение составляют программы для ЭВМ и базы данных, в которых содержатся сведения, составляющие государственную тайну.

Предусмотренная регистрация не является правообразующей и носит факультативный характер, т. е. с ней не связано возникновение прав на программу для ЭВМ или базу данных, однако такая процедура представляется полезной по следующим соображениям.

1. Она является **официальным уведомлением общественности** о наличии у правообладателей прав в отношении рассматриваемых объектов. Подобное уведомление обеспечивается публикацией соответствующих сведений в Официальном бюллетене федерального органа исполнительной власти по интеллектуальной собственности.
2. Государственная регистрация **содействует защите прав** в случае возникновения конфликтных ситуаций при нарушении прав или установлении приоритета. При этом депонированные в федеральном органе материалы могут рассматриваться судом при сборе доказательств в качестве свидетельства наличия соответствующих прав у лица, подавшего заявку на регистрацию сведений.
3. Как свидетельствует практика, факт регистрации программы для ЭВМ или базы данных является для соответствующих органов достаточным основанием для разрешения вывоза данных объектов с территории России. Кроме того, налоговые органы и некоторые страховые компании, занимающиеся страхованием сделок, связанных с объектами интеллектуальной собственности, требуют от своих клиентов предварительной регистрации программ для ЭВМ и баз данных. Официальные документы о регистрации обеспечивают также документальное подтверждение наличия и использования таких объектов для бухгалтерского учета при продаже-покупке лицензий, выплате авторских вознаграждений.

Процедура официальной регистрации программ для ЭВМ и баз данных в целом определена ст. 1262 ГК РФ Ч.4 и включает подачу заявки в федеральный орган исполнительной власти по интеллектуальной собственности (Роспатент), проверку поданных документов и собственно регистрацию. После поступления заявки на регистрацию в Роспатент проверяется наличие необходимых документов и их соответствие установленным требованиям. При положительном результате проверки сведения о программе для ЭВМ или базе данных вносятся, соответственно, в Реестр программ для ЭВМ или Реестр баз данных под уникальным регистрационным номером и выдается заявителю (здесь заявителем называют правообладателя, подавшего заявку на регистрацию программы или базы данных в Роспатент) **свидетельство** о государственной регистрации установленной формы, в котором указаны регистрационный номер объекта по Реестру, название программы или базы данных, имя или наименование правообладателя, фамилии авторов и дата регистрации. Сведения о зарегистрированных программах для ЭВМ и базах данных публикуются в официальном бюллетене Роспатента.

Состав заявки на официальную регистрацию программы для ЭВМ или базы данных (далее - Заявка) определен п. 2 ст. 1262 ГК РФ Ч.1, а также в **Правилах составления, подачи и рассмотрения заявок на официальную регистрацию программ для электронных вычислительных машин и баз данных** (далее - Правила).

Заявка должна относиться к одной программе или одной базе данных. При этом *"Программа для ЭВМ, состоящая из нескольких программ для ЭВМ (программный комплекс), которые не могут быть использованы самостоятельно, регистрируется в целом (без регистрации каждой входящей в нее (него) программы для ЭВМ)"*. Заявка должна содержать следующие **документы**:

1. заявление о государственной регистрации;
2. депонируемые материалы, идентифицирующие программу для ЭВМ или базу данных, включая реферат;
3. документ, подтверждающий уплату государственной пошлины в установленном размере или основание для освобождения от уплаты государственной пошлины или уменьшения его размера.

Далее представлены документы, необходимые для официальной регистрации программы для ЭВМ «комплексная система защиты сайта Protect».

|  |
| --- |
| **Форма РП**  **№ № Регистра-**  **Входящийционный**  **от\_\_\_\_\_\_\_ \_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_ г. от\_\_\_\_\_\_\_ \_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_ г.**  **В ФЕДЕРАЛЬНЫЙ ОРГАН ИСПОЛНИТЕЛЬНОЙ ВЛАСТИ**  **ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ**  **Бережковская наб., 30, корп. 1, Москва, Г-59, ГСП-5, 123995**  **ЗАЯВЛЕНИЕ**  **НА ГОСУДАРСТВЕННУЮ РЕГИСТРАЦИЮ**  **☒ПРОГРАММЫ ДЛЯ ЭВМ или**  **БАЗЫ ДАННЫХ (отметить [X])**  **Представляя указанные ниже документы, подтверждаю (ем) отсутствие сведений, составляющих**  **государственную тайну, и прошу (просим) зарегистрировать программу для ЭВМ (базу данных)** |
| **1. ПРАВООБЛАДАТЕЛЬ (ЗАЯВИТЕЛЬ(И)ОГРН: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  *(для правообладателя- юридического лица)*  **Кудрявцев Иван Владиславович**  **Указанное лицо является:  государственным заказчиком;  муниципальным заказчиком;**  **исполнителем работ по****государственному контракту****муниципальному контракту**  **заказчик работ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  *(наименование заказчика)*  **контракт от\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ № \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **(Указывается полное имя или наименование заявителя(ей) и его (их) место жительства или место нахождения,**  **включая указание страны. Данные о местожительстве автора(ов)-заявителя(ей) приводятся в графе 9А)(Всего заявителей 1 )** |
| **2. ОСНОВАНИЯ ВОЗНИКНОВЕНИЯ ПРАВ НА РЕГИСТРИРУЕМУЮ ПРОГРАММУ ДЛЯ ЭВМ ИЛИ БАЗУ ДАННЫХ**  **(отметить [X]) (заполняется, если заявитель является юридическим лицом, или состав заявителей не соответствует составу авторов)**  **заявитель является раб****отодателем автора  передача прав автором или его правопреемником заявителю**  **передача прав работодателем заявителю**  **право наследования \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** |
| **3. НАЗВАНИЕ РЕГИСТРИРУЕМОЙ ПРОГРАММЫ ДЛЯ ЭВМ ИЛИ БАЗЫ ДАННЫХ**  комплексная система защиты сайта Protect |
| **3А. ПРЕДЫДУЩЕЕ ИЛИ АЛЬТЕРНАТИВНОЕ НАЗВАНИЕ (подчеркнуть)(в свидетельстве не указывается)**  нет |
| **4. НАЗВАНИЕ СОСТАВНОГО ПРОИЗВЕДЕНИЯ (если регистрируемая программа для ЭВМ или база данных является частью составного произведения)**  нет |
| **5. СВЕДЕНИЯ О ПРЕДЫДУЩЕЙ РЕГИСТРАЦИИ** нет  **Номер предыдущей регистрацииДата предыдущей регистрациичисломесяцгод** |
| **6. ДАТА СОЗДАНИЯ РЕГИСТРИРУЕМОЙ ПРОГРАММЫ ДЛЯ ЭВМ ИЛИ БАЗЫ ДАННЫХ (допускается указание только**  **года создания регистрируемой программы для ЭВМ или базы данных)**  **числомесяц2010год** |
| **7.МЕСТО И ДАТА ПЕРВОГО ВЫПУСКА В СВЕТ РЕГИСТРИРУЕМОЙ ПРОГРАММЫ ДЛЯ ЭВМ ИЛИ БАЗЫ ДАННЫХ**  **страна** Российская Федерация **11число02 месяц 2011год** |
| **8. СВЕДЕНИЯ О ПРОИЗВЕДЕНИЯХ, ЯВЛЯЮЩИХСЯ ОБЪЕКТАМИ АВТОРСКОГО ПРАВА(использованных при создании**  **регистрируемой программы для ЭВМ или базы данных)**  PHP |

|  |  |
| --- | --- |
| **9. АВТОРЫ: Всего авторов\_\_1\_\_\_** **авторы отказались быть упомянутыми в качестве таковых** | |
| **9А. СВЕДЕНИЯ ОБ АВТОРЕ ((для указания сведений о других авторах необходимо заполнение формы РП/Доп)**  **Фамилия Имя Отчество:**Кудрявцев Иван Владиславович  **Дата рождения 18число 04месяц 1988год** | |
| **Место жительства, включая указание страны, телефон:**  197343, г. Санкт-Петербург,Торжковская ул., д. 15, кв. 69; тел. 8(964) 375-84-80 | |
| **Краткое описание творческого вклада автора при создании регистрируемой программы для ЭВМ или базы данных:**  Программирование и отладка программ комплекса | |
| **При публикации сведений о государственной регистрации программы для ЭВМ или базы данных прошу: (отметить [X])**  **☒упоминать меня как автора под своим именем** **не упоминать меня как автора (анонимно)**  **упоминать меня как автора под псевдонимом** | |
| **Приведённые сведения верны. Автор:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_**Шустров А.Б. **\_\_\_\_\_\_\_\_\_\_\_\_\_**  **подпись Фамилия И. О.** | |
| **10. СВЕДЕНИЯ О СОДЕРЖАЩИХСЯ В ЗАЯВКЕ ДОКУМЕНТАХ(отметить [X])**  **☒идентифицирующие материалы в формераспечатки исходного текста программы для ЭВМ на3л. в 1 экз.**  **идентифицирующие материалы в иной форме нал. в 1 экз.**  **материалы аудиовизуальных отображений, порождаемых программой для ЭВМ, нал. в 1 экз.**  **материалы, идентифицирующие базу данных, нал. в 1 экз.**  **☒реферат на 1 л. в 2 экз.  иные материалынал. в 1 экз.**  **дополнение к заявлению (форма РП/Доп) нал.вэкз.** **доверенность(и). нал. в 1 экз.**  **☒документ об уплате государственной пошлины на 1 л. в 1 экз.**  **документ, подтверждающий наличие оснований для освобождения от уплаты государственной пошлины, либо уменьшения ее размера, либо отсрочки ее уплаты нал. в 1 экз.**  **документ об уплате тарифа нал. в 1 экз.**  **ходатайствона л. в 1 экз.**  **иные документынал. в 1 экз.** | |
| **11. АДРЕС ДЛЯ ПЕРЕПИСКИ(включая почтовый индекс):** 197343, г. Санкт-Петербург,  Торжковская ул., д.15, кв. 69  **Адресат (включая указание Фамилии И.О. получателя):**Кудрявцев Иван Владиславович  Адрес электронной почты: Номер телефона: 8(964) 375-84-80 | |
| **12. КОНТАКТНЫЕ РЕКВИЗИТЫ ДЛЯ ПРЕДСТАВЛЕНИЯ ТРЕТЬИМ ЛИЦАМ:** 8(964) 375-84-80  **(тел., адрес электронной почты , адрес и др. указываются по желанию правообладателя)** | |
| **ПОДПИСЬ(И) ЗАЯВИТЕЛЯ(ЕЙ) ИЛИ ЕГО(ИХ) ПРЕДСТАВИТЕЛЯ**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Шустров А.Б.**\_\_\_\_**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**«11» \_\_\_02\_\_\_ 2011**\_**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **(От имени юридического лица заявление подписывает руководитель с указанием должности или иное лицо, уполномоченное на это учредительными документами, подпись заверяется печатью юридического лица. Заявление может подписать лицо, уполномоченное на это по доверенности. Подпись любого лица должна быть расшифрована и указана дата подписания заявления. Подписи правообладателей, которые не могут быть размещены в форме РП, размещаются в форме РП/ДОП.)** | |

**ПРОГРАММА ДЛЯ ЭВМ**

**комплексная система защиты сайта Protect**

Фрагменты исходного текста программы

Всего 6 листов

Правообладатель: Кудрявцев Иван Владиславович

Автор: Кудрявцев Иван Владиславович

(ф., и., о.)

(C) Шустров А.Б., 2011

Санкт-Петербург

2011

**СОСТАВ ПРОГРАММНОГО КОМПЛЕКСА**

|  |  |  |
| --- | --- | --- |
| protect.php | - | Основной модуль библиотеки; |
| sd\_captcha.php | - | Модуль капчи; |
| shrift | - | Папка со шрифтами для капчи; |

**Р Е Ф Е Р А Т**

Автор: Кудрявцев Иван Владиславович

Правообладатель: Кудрявцев Иван Владиславович

Программа для ЭВМ: комплексная система защиты сайта Protect

Аннотация: Программный комплекс предназначен для защиты сайта от SQL-атак, XSS-атак, ботов методом капча, PHP-инъекций. Каждый программный модуль комплекса (подпрограмма) выполняет определенный этап комплексной защиты сайта. Модуль sd\_captcha.php отвечает за вывод в браузер случайно сгенерированного кода, который легко распознается человеком, но не поддается расшифровке машинной. Модуль protect.php проверяет введенный человеком код на соответствие истине, а также фильтрует другие данные введенные пользователем.

Тип ЭВМ: Рабочие станции, Персональные ЭВМ (ПЭВМ)

ОС: WindowsXP/7/VISTA

Язык программирования: PHP

Объем: 445 Кбайт

**ЛИЦЕНЗИОННЫЙ ДОГОВОР**

**НА ИСПОЛЬЗОВАНИЕ ПРОГРАММЫ ДЛЯ ЭВМ**

**Стороны в Договоре:**

Гражданин Кудрявцев Иван Владиславович, проживающий по адресу: 197343, г. Санкт-Петербург, ул. Торжковская, д. 15, кв. 69, именуемый в дальнейшем "ЛИЦЕНЗИАР", с одной стороны, и

Государственное образовательное учреждение высшего профессионального образования "Санкт-Петербургский государственный электротехнический университет "ЛЭТИ" им. В.И.Ульянова (Ленина)", именуемый в дальнейшем "ЛИЦЕНЗИАТ", в лице проректора по научной работе деятельности Шестопалова М.Ю., действующего на основании Доверенности, с другой стороны,

**принимая во внимание:**

1. что Лицензиар является автором и правообладателем программы для ЭВМ "комплексная система защиты сайта Protect ";
2. Лицензиат желает получить на условиях настоящего Договора лицензию на использование упомянутой программы для ЭВМ.;
3. Лицензиар готов предоставить Лицензиату такую лицензию,

**договорились о следующем.**

**1. Термины и их определения**

1. "ПРОГРАММА ДЛЯ ЭВМ (ПрЭВМ)" - программное обеспечение " комплексная система защиты сайта Protect".
2. "ДОКУМЕНТАЦИЯ" - комплект документов, передаваемых Лицензиаром Лицензиату, включающий руководство пользователя по применению и обслуживанию программы для ЭВМ.
3. "ПРОИЗВОДСТВЕННАЯ ПЛОЩАДКА" - научные лаборатории и кафедры Лицензиата.
4. "РАБОЧЕЕ МЕСТО" - конкретная ЭВМ, на которой используется Программа для ЭВМ.

**2. Предмет Договора**

1. Лицензиар предоставляет Лицензиату на срок действия настоящего Договора и за вознаграждение, уплачиваемое Лицензиатом, неисключительную лицензию на использование ПрЭВМ. При этом Лицензиату предоставляется право на установку ПрЭВМ не более чем на 10 (десяти) Рабочих местах.
2. Лицензиар передает Лицензиату Документацию к ПрЭВМ.
3. Предоставленное Лицензиату в рамках настоящего Договора право ограничено Производственной площадкой.
4. Лицензиар осуществляет авторский контроль за соблюдением объемов использования ПрЭВМ по настоящему Договору, при этом Лицензиат обеспечивает возможность такого контроля.
5. Лицензиар сохраняет за собой право самому использовать ПрЭВМ и предоставлять неисключительные лицензии на право ее использования третьим лицам.

**3. Обеспечение Договора**

1. Лицензиар передает Лицензиату ПрЭВМ в объеме и виде, достаточном для ее использования, и Документацию в течение 15 (пятнадцати) дней со дня подписания настоящего Договора. ПрЭВМ передается Лицензиату в виде комплекта дискет в количестве 5 (пяти) штук, содержащих ПрЭВМ. По факту передачи ПрЭВМ и Документации составляется акт сдачи-приемки с перечнем переданных материалов, подписываемый обеими Сторонами.
2. Если Лицензиат установит неполноту или неправильность полученных ПрЭВМ или Документации, то Лицензиар в течение 15 (пятнадцати) дней после сообщения ему об этом Лицензиатом обязан передать недостающие материалы или устранить недостатки ранее переданных ПрЭВМ и Документации.
3. Для оказания помощи в освоении ПрЭВМ Лицензиар по просьбе Лицензиата оказывает консультации пользователям ПрЭВМ.
4. Для целей использования ПрЭВМ в объеме, предусмотренном п. 2.1 настоящего Договора, Лицензиат может изготавливать в необходимом ему количестве копии ПрЭВМ и копии Документации.

**4. Усовершенствования**

1. Лицензиар обязуется незамедлительно информировать Лицензиата о всех произведенных им усовершенствованиях ПрЭВМ и, при желании Лицензиата, передать ему в согласованные сроки новые варианты ПрЭВМ. В отношении новых вариантов ПрЭВМ, переданных Лицензиаром Лицензиату, распространяются все условия настоящего Договора.
2. Лицензиат обязуется предоставлять Лицензиару информацию об использовании ПрЭВМ, которая могла бы быть полезной для усовершенствования ПрЭВМ.

**5. Платежи**

1. За предоставление прав, предусмотренных настоящим Договором, Лицензиат выплачивает Лицензиару единовременное вознаграждение в размере 20000(двадцать тысяч) рублей.
2. Вознаграждение, предусмотренное п. 5.1 настоящего Договора, выплачивается Лицензиатом в течение 30 (тридцати) дней, следующих после подписания акта приемки-сдачи.

**6. Реклама**

1. Лицензиат обязуется при опубликовании результатов исследований, полученных с использованием ПрЭВМ, сообщать в рекламных целях, что исследования производились с использованием ПрЭВМ Лицензиара с указанием авторского права Лицензиара.

**7. Защита передаваемых прав**

1. Лицензиат обязуется не вносить самовольно каких-либо изменений в ПрЭВМ и Документацию и не дополнять их какими-либо комментариями. Подобные изменения или дополнения возможны только с согласия Лицензиара.
2. Лицензиат обязуется предпринимать все необходимые меры для предотвращения несанкционированного копирования ПрЭВМ и Документации третьими лицами, а также несанкционированной передачи ПрЭВМ и Документации работниками Лицензиата третьим лицам.
3. Если Лицензиату станет известно о противоправном использовании ПрЭВМ третьими лицами, то он незамедлительно сообщит об этом Лицензиару.

**8. Ответственность Сторон и разрешение споров**

1. За невыполнение или ненадлежащее выполнение обязательств по настоящему Договору Стороны несут имущественную ответственность в соответствии с действующим законодательством.
2. Стороны освобождаются от ответственности за неисполнение или ненадлежащее исполнение обязательств, принятых по настоящему Договору, если неисполнение явилось следствием обстоятельств непреодолимой силы (форс-мажор).
3. Сторона, нарушившая свои обязательства по настоящему Договору, освобождается от ответственности за неисполнение или ненадлежащее исполнение этих обязательств, если это нарушение было вызвано причинами, за которые отвечает другая Сторона.
4. В случае возникновения споров между Лицензиаром и Лицензиатом по вопросам, предусмотренным настоящим Договором, Стороны примут все меры к разрешению их путем переговоров между собой. В случае невозможности разрешения указанных споров путем переговоров они будут разрешаться в порядке, предусмотренном действующим законодательством.

**9. Срок действия Договора и условия его расторжения**

1. Настоящий Договор заключен на срок 2 года и вступает в силу с даты его подписания обеими Сторонами.
2. По истечении срока действия настоящего Договора Лицензиат вправе использовать ПрЭВМ, включая усовершенствованные варианты, на Производственной площадке на любом количестве Рабочих мест. При этом обязательства Лицензиата, предусмотренные пп. 7.1 и 7.2 настоящего Договора, сохраняются бессрочно.
3. Действие настоящего Договора по обоюдному согласию Сторон может быть досрочно прекращено, но не ранее чем через три месяца после предложения об этом одной из Сторон. При этом Лицензиат не освобождается от обязательств по платежам, возникшим до расторжения настоящего Договора.
4. Настоящий Договор может быть досрочно расторгнут в одностороннем порядке со стороны Лицензиара из-за невыполнения Лицензиатом своих обязательств по пп. 7.1 или 7.2. В этом случае Лицензиат лишается права дальнейшего использования ПрЭВМ в любой форме и обязан вернуть ее Лицензиару.
5. Если Лицензиат откажется от дальнейшего использования ПрЭВМ, то он уничтожит все имеющиеся у него копии ПрЭВМ.

**10. Заключительные положения**

1. Все изменения и дополнения к настоящему Договору действительны только в тех случаях, если они совершены в письменной форме и подписаны обеими Сторонами.
2. Стороны не имеют права передавать свои права и обязательства по настоящему Договору третьим лицам без письменного согласия на то другой Стороны.
3. Во всем остальном, что не предусмотрено условиями настоящего Договора, будут применяться нормы законодательства Российской Федерации.

**11. Адреса Сторон**

1. ЛИЦЕНЗИАР: Иванов Сергей Петрович, адрес: 190000, Санкт-Петербург, ул. Почтамтская, д. 1, кв. 10.
2. ЛИЦЕНЗИАТ: СПбГЭТУ, адрес: 197376, Санкт-Петербург, ул. Проф. Попова, д. 5.

Настоящий Договор составлен в двух экземплярах для каждой из Сторон и подписан "\_\_\_"\_\_\_\_\_\_\_\_\_\_ 200\_\_ г. в г. Санкт-Петербурге.

|  |  |
| --- | --- |
| **ЛИЦЕНЗИАР:** | **От ЛИЦЕНЗИАТА:** |
| \_\_\_\_\_\_\_\_\_\_\_ А.В. Саянц | Проректор по научной работе СПбГЭТУ  \_\_\_\_\_\_\_\_\_\_\_\_ М.Ю.Шестопалов |

СПИСОК ЛИТЕРАТУРЫ

* + 1. Технико-экономическое обоснование дипломных проектов. Под ред. Проф. Беклкшова В.К., М. Высшая школа, 1999. – 65 с.
    2. С.В. Маклаков. CASE – средства разработки информационных систем.- М.: Диалог-Мифи, 2000. – 120 c.
    3. С.А. Орлов. Технологии разработки программного обеспечения. СПб.: Питер, 2002. – 463 c.
    4. Д. Н. Колисниченко - Самоучитель PHP 5.
    5. <http://www.php.su/> - PHP, MySQL и другие веб-технологии.
    6. <http://www.php.ru/> - форум PHP программистов.

ПРИЛОЖЕНИЕ

Листинг основного модуля библиотеки **protect.php**

<?php

// \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

// Модуль: protect.php Защита от SQL-атак, XSS-атак, и ботов(методом "капча")

// Дата создания: январь 2011 года

// Автор: Саянц Антон

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//Ведите сюда ваши переменные, для дальнейшей работы библиотеки. По умолчанию, все функции отключены, длявключение замените значения "vykl" на свои данные.

$namik ="vykl"; // переменная для проверки имени(логина) или пароля. Любого поля типа text

$famik = "vykl"; // переменная для проверки Фамилии или пароля. Любого поля типа text

$mailik = "vykl"; // переменная для проверки e-mail(адреса электронной почты).

$textik = "vykl"; // переменная для проверки общего текстового поля (textarea).

$textik1 = "vykl"; //переменная для проверки общего текстового поля (textarea). Преобразует специальные символы в HTML сущности, для безопасного их отображения на странице.

$teg = '<i><b>'; //перечислите теги, которые вы не хотите удалять из текстового поля.

$znakl = array("<",">","`","~","'","\"","\\","?","&","$","^","\*","{","}","[","]","(",")","%",";",":","/","!"); //массив вырезаемых из $namik символов

$znakt = array("`","~","'","\"","\\","$","^","\*","{","}","[","]"); //массив вырезаемых из $textik и $textik1 символов

$simn = 20; //кол-во символов вырезаемое из передаваемых данных в $namik, с начала.

$simm = 30; //кол-во символов вырезаемое из передаваемых данных в $mailik, с начала.

$simt = 300; //кол-во символов вырезаемое из передаваемых данных в $textik и $textik1, с начала.

$provses = "vykl"; //переменная для для проверки переданного от пользователя значения капчи.Возвращает "2", если капча верна. "1" Если не верна.

$cifri = "vykl"; // переменная для для проверки цифрового поля int

$cifrf = "vykl"; // переменная для для проверки цифрового поля float

//\_\_\_\_\_\_\_\_\_\_\_начинаем работу!\_\_\_\_\_\_\_\_\_\_\_\_\_

if ($provses != "vykl"){

// стартуем сессию

session\_start ();

//Вырезаем нужное кол-во символов капчи

$provses = substr($provses,0,$\_SESSION['cap\_length\_max']);

//Проверяем совпадает ли переданный идентификатор с реальным

//Проверка капчи, функция возвращает true, если данные введны правильно. FALSE в противном случае.

function provsesiya ($provses){

if( $\_SESSION['captcha\_code'] == $provses){

//Разрушаем сессию

session\_destroy();

return true; //капча верна

}

returnfalse;

} }

//проверканапустотуполя

if ($cifri != "vykl"){

settype($cifri, "integer"); // вернет только int значение переменной $cifri или FALSE(0);

}

//проверка на пустоту поля

if ($cifrf != "vykl"){

settype($cifrf, "float"); // вернет только float значение переменной $cifrf или FALSE(0);

}

//проверка на пустоту поля

if ($namik != "vykl"){

//обработака пользовательских данных:

$namik = str\_replace($znakl,"",strip\_tags(stripslashes(substr($namik,0,$simn))));

}

//проверка на пустоту поля

if ($famik != "vykl"){

//обработака пользовательских данных:

$famik = str\_replace($znakl,"",strip\_tags(stripslashes(substr($famik,0,$simn))));

}

//проверка на пустоту поля

if ($mailik != "vykl"){

//обработака пользовательских данных:

$mailik = strip\_tags(stripslashes(substr($mailik,0,$simm)));

// Проверка корректности адреса e-mail, функция возвращает true, если данные соответствуют регулярному выражению и кол-во символов больше 8.

functioncheck\_email ($mailik)

{

if(preg\_match('|^[-0-9A-Za-z\_\.]+@[-0-9A-Za-z^\.]+\.[a-z]{2,6}$|i',$mailik) andstrlen($mailik)>8) //регулярное выражение для проверки почты

{ return true; }

return false;

}

}

if($textik != "vykl") {

//обработакапользовательскихданных:

$textik = quotemeta(str\_replace($znakt, "",str\_replace/\*заменяемсимволывводана<br>\*/(array("\r\n","\r","\n"), "<br/>", strip\_tags(stripslashes(substr($textik,0,$simt)),$teg))));

}

if ($textik1 != "vykl"){

//обработака пользовательских данных:

$textik1 = quotemeta(str\_replace($znakt, "",str\_replace/\*заменяем символы ввода на <br>\*/(array("\r\n","\r","\n"), "<br/>", htmlspecialchars(strip\_tags(stripslashes(substr($textik1,0,$simt)),$teg)))));

}

//\_\_\_\_\_\_Проверка пользовательских данных закончена, можно приступать к работе с ними!\_\_\_\_

?>

Листингмодулякапчи**sd\_captcha.php**

<?php

session\_start();

$cap\_width=120;// ширинакапчи

$cap\_height=40;// высота капчи

$cap\_quality=10;// качество изображения, 100 = 100 процентов качество

$cap\_length\_min=4;// минимальное кол-во символов в коде

$cap\_length\_max=5;// максимальное кол-во символов в коде

$cap\_digital=1;// true (1) - в коде разрешено использовать цифры

$cap\_latin\_char=1;// true (1) - в коде разрешено использовать латинские символы

$cap\_russian\_char=0;// true (1) - в коде разрешено использовать русские символы

// \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

function code\_generic($\_length,$\_digital=1,$\_latin\_char=1,$\_russian\_char=0)// генерацияпароля

// \_legnth - длина пароля

// \_digital - 1 разрешена генерация цифр в пароле

// \_latin\_char - 1 разрешена генерация латинских букв в пароле

// \_russian\_char - 1 разрешена генерация русский букв в пароле

{

$dig=array(0,1,2,3,4,5,6,7,8,9);// массив с цифрами

// массив с русскими буквами

$rus=array('а','б','в','г','д','е','ж','з','и','к','л','м','н','о',

'п','р','с','т','у','ф','х','ц','ч','ш','щ','ь','ы','ъ','э','ю','я','ё','й');

// массив с лотинскими буквами

$lat=array('a','b','c','d','e','f','g','h','j','k','l','m','n','o',

'p','q','r','s','t','u','v','w','x','y','z');

$main=array();

// если требуется объединяем массив из цифр

if ($\_digital) $main=array\_merge($main,$dig);

// если требуется объедениям массив из латинских буковок

if ($\_latin\_char) $main=array\_merge($main,$lat);

// если требуется объеденяем массив из русских буковок

if ($\_russian\_char) $main=array\_merge($main,$rus);

shuffle($main);// перемешать массив в случайном порядке

// вырезаем нужное количество символов из перемешенного массива, и аля паролька готова ;)

$pass=substr(implode('',$main),0,$\_length);

return $pass;// возвращаемрезультат

}

// \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

$l=rand($cap\_length\_min,$cap\_length\_max);// определяемкол-восимволоввкоде

$code=code\_generic($l,$cap\_digital,$cap\_latin\_char,$cap\_russian\_char);// генерируемкод

$\_SESSION['captcha\_code']=$code;// запоминаем код для будущей сверки

$\_SESSION['cap\_length\_max']=$cap\_length\_max;//запоминаем макс. кол-во символов капчи

$canvas=imagecreatetruecolor($cap\_width,$cap\_height);// создаем прямоугольник куда и будем рисовать

$c=imagecolorallocate($canvas,rand(230,250),250,250);// цветфоназаполнения

imagefilledrectangle($canvas,0,0,$cap\_width,$cap\_height,$c);// заливаемкартинкуфоновымцветом

$count=strlen($code);// узнаемдлинукода

$color\_text=imagecolorallocate($canvas,rand(20,150),rand(20,150),190); //цветтекста

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_массивсошрифтами\_\_\_\_\_\_\_\_\_\_\_\_\_\_

$shrift[0] = "shrift/a\_CampusGrav.ttf" ;

$shrift[1] = "shrift/download\_fonts\_ttf-rus\_CampusOtl\_Bold.ttf" ;

$shrift[2] = "shrift/download\_fonts\_ttf-rus\_Bookman\_Old\_Style.ttf" ;

$shrift[3] = "shrift/download\_fonts\_ttf-rus\_BoisterBlack.ttf" ;

$shrift[4] = "shrift/download\_fonts\_ttf-rus\_TrooverRoman.ttf" ;

$shrift[5] = "shrift/ELEPHNT.ttf" ;

$shrift[6] = "shrift/IMPRISHA.ttf" ;

//выбор шрифта

$sh = rand(0,6);

//создаём шум на фоне в виде коротеньких линий

for ($i=0; $i<=80; $i++) {

$xn = rand(2,$cap\_width);

$yn = rand(2,$cap\_height);

$xk = $xn + rand(-10,10);

$yk = $yn + rand(-10,10);

imageline($canvas,$xn,$yn,$xk,$yk,$color\_text);

}

for($it=0;$it<$count;$it++)// прорисуемкаждуюбуковку

{ $letter=$code[$it];// берембуковку

//imagestring($canvas,6,(10\*$it+20),$cap\_height/4,$letter,$color\_text);

imagettftext($canvas,20,rand(-35,35),(18\*$it+rand(10,12)),$cap\_height/1.5,$color\_text,$shrift[$sh],$letter);

}

// создаем соответствующий заголовок для корректности вывода изображения

// делаем все так, чтобы браузеры не кэшировали нашу картинку

header('Expires: Mon, 26 Jul 1997 05:00:00 GMT');

header('Cache-Control: no-store, no-cache, must-revalidate');

header('Cache-Control: post-check=0, pre-check=0',false);

header('Pragma: no-cache');

header('Content-Type: image/jpeg');

imagejpeg($canvas,null,$cap\_quality);// непосредственно создаем картинку

?>