

**Протокол имплементации**

**Интернетный ресурс базы данных   
речных и озерных рыб водоемов РФ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Проектная группа | -- | | | |
| Заказчик | Самостоятельный проект | | | |
| Проект | Концепция интернетного ресурса свободного доступа к ограниченной базе данных речных и озерных рыб водоемов РФ | | | |
| Руководитель проекта | Артур Мангус | | | |
| Участники проекта |  | | | |
| Документ | Протокол имплементации | | Количество страниц |  |
| Автор документа | Артур Мангус | | | |
| Создан | 18.12.2022 | | | |
| Последнее изменение | 18.12.2022 | | | |
| Статус обработки | X | В обработке  Представлен разработчикам и закзчику  Одобрен  Закрыт | | |

**История документа**

|  |  |  |  |
| --- | --- | --- | --- |
| **Версия** | **Дата** | **Автор изменения** | **Описание / замечание** |
| 0.1 | 18.12.2022 | Артур Мангус |  |

Оглавление

[1 Общее описание 3](#_Toc122271935)

[2 Макет с использованием CSS 3](#_Toc122271936)

[3 Cookie и JavaScript 5](#_Toc122271937)

[4 Реализация меню DropDown в CSS 5](#_Toc122271938)

[5 Реализация базы данных MySQL 6](#_Toc122271939)

[6 Работа с базой данных из PHP 6](#_Toc122271940)

[Приложения и ресурсы 6](#_Toc122271941)

Настоящий протокол направлен на анализ реализации приложения, с целью улучшения кода в последующих проектах.

# Общее описание

При реализации файла index.php и UpdatePage.php было принято очень верное решение использовать html\_entity\_decode всего один раз, пропуская через него весь код страницы.   
Текст index.php получился очень компактным и представляет из себя только три блока:

* Включение скриптов CSS и JavaScript
* Описание заголовка и мета данных
* HTML код страницы

Текст UpdatePage.php также очень компактный и включает в себя два блока:

* Обработка событий массивов $\_GET и присвоение значений массива $\_COOKIE
* HTML код страницы

Так как технология PHP предполагает создание страниц на стороне сервера, который в свою очередь возвращает полный код в формате HTML, для каждой отдельно сгенерированной страницы, то нет необходимости следовать каким-то особым требованиям в формировании кода страницы.

Для облегчения понимания кода был создан класс PageSubject, сводящий фрагменты макета.

Все классы, возвращающие код макета HTML, реализованы как статические. Использование динамических классов в данном случае не оправдано, так как размер используемых ресурсов памяти на стороне сервера для данной реализации ничтожен. В свою очередь вызов статических методов делает код более компактным и читаемым.

Реализация проводится в IDE PhpStorm. Программа автоматически переформатирует табуляцию, позволяющую лучше проанализировать глубину вложения. Для избегания автоматического форматирования кода необходимо встраивать после объявления класса строку с комментарием // @formatter:off

При тестировании страницы в браузере необходимо каждый раз вызывать обновление клавиатурным сокращением Ctrl+F5.

# Макет с использованием CSS

Реализация скриптов CSS основана на рекомендациях книги Макфарланда «Новая большая книга CSS» издательства O’Railly.

Основной макет построен на блоках div. Блоки выстраиваются автоматически по вертикали образуя макет QLinearVBox. Для организации блоков по горизонтали используется свойство display со значением flex. Необходимо проверить наследуется ли это свойство вложенными блоками. Если наследуется, то его необходимо переопределять со значением initial.

Для установки значений по умолчанию использован файл Main.css, в котором определены базовые установки для всех элементов макета. Так, например, для предотвращения копирования текста использован следующий код:

|  |
| --- |
| -webkit-touch-callout: none; */\* iOS Safari \*/* -webkit-user-select: none; */\* Safari \*/* -khtml-user-select: none; */\* Konqueror HTML \*/* -moz-user-select: none; */\* Old versions of Firefox \*/* -ms-user-select: none; */\* Internet Explorer/Edge \*/* user-select: none; */\*generic\*/* |

Для предотвращения нарушений цветового представления из-за специфических настроек браузера использована установка белого цвета фона по умолчанию:

|  |
| --- |
| body {  line-height: 1.2;  background-color: white; } |

Для выравнивания блоков по вертикали, относительно родительского элемента макета, использован следующий код:

|  |
| --- |
| position: relative; margin: auto; top: 0; right: 0; bottom: 0; left: 0; |

Для использования шрифтов, не включенных в список стандартных, используется файл Font.css с определением шрифтов. Файл импортируется в другие файлы css вызовом @import "Fonts.css";. Строки инициализации font-family: "Roboto Condensed"; font-size: 100%; включены в файл Main.css для всех элементов макета по умолчанию. Таким образом необходимо только менять стиль и размер шрифта. Для определения пользовательских шрифтов в файле Fonts.css использован следующий код:

|  |
| --- |
| @font-face {  font-family: 'Roboto Condensed';  src: url('../fonts/RobotoCondensed-Regular.eot');  src: local('Roboto Condensed'), local('RobotoCondensed-Regular'),  url('../fonts/RobotoCondensed-Regular.eot?#iefix') format('embedded-opentype'),  url('../fonts/RobotoCondensed-Regular.woff') format('woff'),  url('../fonts/RobotoCondensed-Regular.ttf') format('truetype');  font-weight: normal;  font-style: normal; } |

Для работы с изображениями использован рекомендуемый код:

|  |
| --- |
| background: url("../img/headerkontakt.jpg") no-repeat center center; |

При разработке кнопок с использованием файлов svg необходимо учитывать, что размер рабочей области документа изображения должен соответствовать конечному необходимому размеру, используемому в приложении. Все графические элементы изображения должны лежать в рабочей области документа. Для использования разных состояний иконки необходимо каждое из этих состояний сохранить в файле svg в определенных координатах. Состояние может быть определено классом class=“‘.$state.‘“ или подклассом .class:hover. Смена состояния осуществляется смещением координат:

|  |
| --- |
| background-position: left 0px top -38px; |

При работе с селекторами была отмечена значительная разница использования слитного и раздельного написания селекторов классов. Так слитное написание относится к самому элементу, при необходимости более точной идентификации (например обращение к общему классу подклассу и идентификатору .class.subclass#id), а раздельное обращается к дочерним элементам (например класс родителя класс дочернего элемента .parentClass .childClass). Свойство position со значением relativ задает координаты относительно родительского компонента. Значение absolut задает координаты относительно окна браузера.

При использовании рамок вокруг элементов макета необходимо не забывать добавлять свойство border-style: solid;.

# Cookie и JavaScript

Для запоминания выбора пользователя, использован массив глобальных переменных $\_COOCKIE. Все кнопки обрабатывают событие onClick=““ в файле NavicationControl.js. Запрос формируется из значений, полученных из глобальных переменных следующим образом (все строки кода релевантны):

|  |
| --- |
| function *setFamilyId*(control) {  if (document.cookie.length > 0) {   var contentID = *getCookieById*("contentID") + '&';  var pageID = *getCookieById*("pageID") + '&';  var viewID = *getCookieById*("viewID") + '&';  var familyID = 'familyID=' + control.getAttribute("name");  document.cookie = familyID;   var requestData = contentID + pageID + viewID + familyID;   var request = new XMLHttpRequest();  request.open('GET', './php/UpdatePage.php?' + requestData, false);  request.send(null);   if (request.status == 200) {  document.getElementById("mainContent").innerHTML = request.responseText;  }  } } |

Получение глобальных переменных и их запись в JavaScript работает надежнее чем в php.

# Реализация меню DropDown в CSS

Пример простой реализации меню DropDown:

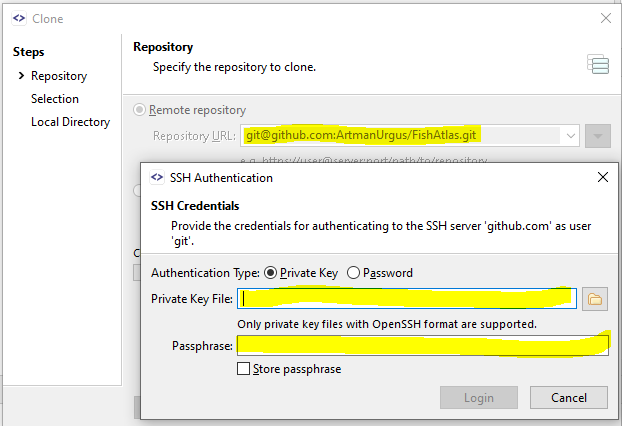
|  |
| --- |
| <style> /\* Style The Dropdown Button \*/ .dropbtn {   background-color: #4CAF50;   color: white;   padding: 16px;   font-size: 16px;   border: none;   cursor: pointer; }  /\* The container <div> - needed to position the dropdown content \*/ .dropdown {   position: relative;   display: inline-block; }  /\* Dropdown Content (Hidden by Default) \*/ .dropdown-content {   display: none;   position: absolute;   background-color: #f9f9f9;   min-width: 160px;   box-shadow: 0px 8px 16px 0px rgba(0,0,0,0.2);   z-index: 1; }  /\* Links inside the dropdown \*/ .dropdown-content a {   color: black;   padding: 12px 16px;   text-decoration: none;   display: block; }  /\* Change color of dropdown links on hover \*/ .dropdown-content a:hover {background-color: #f1f1f1}  /\* Show the dropdown menu on hover \*/ .dropdown:hover .dropdown-content {   display: block; }  /\* Change the background color of the dropdown button when the dropdown content is shown \*/ .dropdown:hover .dropbtn {   background-color: #3e8e41; } </style>  <div class="dropdown">   <button class="dropbtn">Dropdown</button>   <div class="dropdown-content">     <a href="#">Link 1</a>     <a href="#">Link 2</a>     <a href="#">Link 3</a>   </div> </div> |

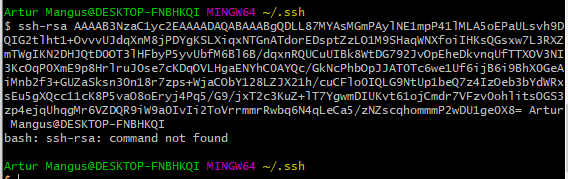
# Реализация базы данных MySQL

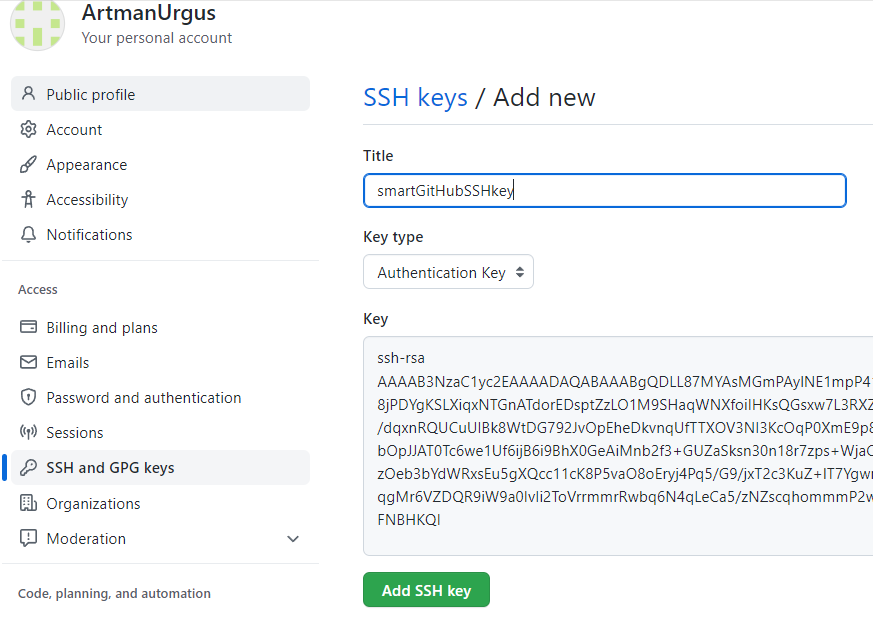
# Работа с базой данных из PHP

# Подключение проекта к GitHub

<https://github.com/ArtmanUrgus/FishAtlas.git>







# Приложения и ресурсы