ГОСУДАРСТВЕННЫЙ КОМИТЕТ СВЯЗИ, ИНФОРМАТИЗАЦИИ И ТЕЛЕКОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ РЕСПУБЛИКИ УЗБЕКСИТАН
ТАШКЕНТСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

К защите.

Зав.кафедрой

_____

_____2014г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

на тему «**Разработка каталога мультимедийных программных ресурсов**»

Выпускник _____ Исматуллаев С. О.

Руководитель _____ Ишанходжаева Г. Р.

Консультант по БЖД _____ Абдуллаева С.М

Рецензент _____ Керимов К.Ф.

Ташкент - 2014

ГОСУДАРСТВЕННЫЙ КОМИТЕТ СВЯЗИ, ИНФОРМАТИЗАЦИИ И ТЕЛЕКОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ РЕСПУБЛИКИ УЗБЕКСИТАН
ТАШКЕНТСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Факультет: Компьютерный инжиниринг
Кафедра: Мультимедийные технологии
Направления (специальность): 5521900 -“Информатика и информационные технологии”

У Т В Е Р Ж Д А Ю
Зав кафедрой_____
«_____» _____2014г.

ЗАДАНИЕ

на выпускную квалификационную работу
Исматуллаева Собиржона Омонжон угли
(фамилия, имя, отчество)

1. Тема работы: *«Разработка каталога мультимедийных программных ресурсов»*
2. Утверждена приказом по университету от 19.04. 2014 г. № 254-15
3. Срок сдачи законченной работы: 31.05.2014 г.
4. Исходные данные к работе: *В ходе разработки каталога мультимедийных программных ресурсов были использованы: техническое задание, материалы из http://wikipedia.org, http://habrahabr.ru/, документация по php.*
5. Содержание расчётно–пояснительной записи (перечень подлежащих разработке вопросов): *Описание предметной области, Проектирование. Взаимодействие информационной системы с пользователями, Разработка алгоритма программы.*
6. Перечень графического материала: блок-схема разработки сайта, функциональная и организационная структуры сайта
7. Дата выдачи задания: 03.03.2014 г.

Руководитель _____ Задание принял _____
(подпись) (подпись)

8. Консультант по отдельным разделам выпускной работы

Раздел	Ф.И.О. руководителя	Подпись дата	
		Задание выдал	Задание получил
Основная часть	Ишанходжаева Г. Р.	05.03.2014 г.	05.03.2014 г.
БЖД	Абдуллаева С.М	20.03.2014 г.	20.03.2014 г.

9. График выполнения работы

№	Наименование раздела работы	Срок выполнения	Отметка руководителя о выполнении
1.	Теоретические основы создания каталога мультимедийных программных ресурсов	05.03.2014-17.03.2014	
2.	Проектирование мультимедийного приложения	18.03.2014- 18.04.2014	
3.	Разработка алгоритма программы «GrowTo»	19.04.2014-19.05.2014	
4.	БЖД	20.03.2014-28.05.2014	
5.	Заключение	29.05.2014-31.05.2014	

Выпускник _____ «____» _____2014 г.

(подпись)

Руководитель_____ «____» _____2014 г.

(подпись)

Данная выпускная квалификационная работа посвящена разработке каталога мультимедийных программных ресурсов, представляющий собой сайт, с помощью которого любой дизанейрский проект разрабатывается в режиме реального времени в on-line в процессе непосредственного общения проектировщика и заказчика. В ходе разработки каталога мультимедийных программных ресурсов был использован гипертекстовый язык разметки HTML (Hyper Text Markup Language), язык программирования JavaScript, CSS (Cascade Style Sheets), система управления базами данных – MySQL, язык создания динамических ресурсов PHP, для дизайна использовались Photoshop, After Effects, Cinema 4D.

Ushbu bitiruv malakaviy ishi ozi bilan klient va dizaynerni aloqasida onlayn tarzda dizayn proyektini yaratishga imkon beruvchi multimedia dasturiy resurslar katalogini yaratishga bagishlangan. Multimedia dasturiy resurslar katalogini yaratish uchun gipertekst belgi tili HTML (Hyper Text Markup Language), JavaScript dasturlash tili, CSS (Cascade Style Sheets), malumotlar bazalarini boshqaruv tizimi MySQL va dinamik resurslar yaratish tili PHP ishlatilgan. Dizayn uchun Photoshop, After Effects, Cinema 4D dasturlardan qo’llanildi.

This graduate work is directed to multimedia program resources catalog development, that allow to clients to interact with designer of this resource for design creating. For this purpose, we have used Hyper Text Markup Language (HTML), JavaScript programming language, Cascade Style Sheets (CSS), DBMS MySQL (Data Base Management System), dynamic resources creating language PHP. For design used Photoshop, After Effects, Cinema 4D.

СОДЕРЖАНИЕ

Введение

В целях регулирования отношений в области информатизации, использования информационных ресурсов и информационных систем, Президент Узбекистана в 2004 году подписал Закон «Об информатизации», результатом которого должно стать создание национальной информационной системы с учетом современных мировых тенденций развития и совершенствования информационных ресурсов, информационных технологий и информационных систем. В связи с этим законом и велением времени необходимо рассмотреть, проанализировать и создать каталог мультимедийных программных ресурсов [1,2].

В условиях глобализации мирового хозяйства различные организации стремятся заявить о себе во всемирной сети; не остаются в стороне и организации и каталоги по разработке мультимедийных программных ресурсов.

В результате анализа существующих каталогов мультимедийных программных ресурсов был выявлен один из наиболее существенных недостатков. Основное внимание сайтов обращено на способы достижения целей - пути создания мультимедийного программного продукта. Несомненно, способ разработки мультимедийного программного продукта очень важен для пользователя, но для пользователя, осмелимся утверждать, первостепенной задачей является поиск информации и ее окончательных вид, т.е. каждый пользователь приблизительно должен видеть готовый мультимедийный программный продукт.

В данной работе каталог больше обращен на доступность информации. В целях улучшения поиска необходимой информации сайт разделен на логические блоки.

Объект исследования – каталог мультимедийных программных ресурсов “GrowTO”.

Актуальность:

В настоящее время информационное мультимедийное пространство охватывает практически все сферы нашей жизнедеятельности. Наиболее актуально использование информационного пространства, развитие разработок 3D-продуктов, а также продуктов, основанных на анимации, для улучшения качества визуальных услуг.

Цель и задачи:

Целью данной выпускной квалификационной работы является разработка каталога мультимедийных программных ресурсов, разработанный с помощью сред программирования HTML, JavaScript, CSS, Photoshop, After Effects, Cinema 4D.

Для достижения поставленной цели были решены следующие задачи:

1. Проведен анализ условий использования Интернет-приложений в мультимедийных процессах.
2. Исследованы имеющиеся аналоги WEB-каталогов разработки мультимедийных программных ресурсов для сайта клиент(заказчик)-исполнитель.
3. Разработана структуры и реализация WEB-приложения с учетом особенностей мультимедийного процесса “GrowTO”.

Инструментом разработки каталога мультимедийных программных ресурсов является CMS-Drupal. Drupal (Друпал) — система управления содержимым, используемая также как каркас для веб-приложений (*CMF*), написанная на языке *PHP* и использующая в качестве хранилища данных реляционную базу данных (поддерживаются *MySQL*, *PostgreSQL* и другие).

Структура работы

- Первая глава «Описание предметной области» посвящена изучению теоретических основ сферы разработки, анализу существующих информационных систем.
- Вторую главу мы посвятили проектированию. Она включает такие этапы,

как «Этапы проектирования», «Взаимодействие пользователя с информационной системой».

- В третьей главе излагается реализация информационной системы, т.е. ее программирование, где описаны два раздела: «коды главных модулей» и «интерфейс».
- В четвертой главе описывается «Безопасность жизнедеятельность».
- В заключении проанализированы теоретические и практические навыки.
- В списке использованной литературы приведен весь список литературы, использованной для выполнения данной квалификационной работы.
- В приложении приведен программный код всего информационного ресурса.

Выпускная квалификационная работа состоит из введения, четырех разделов, заключения, списка литературы, приложения, 16 рисунков, 5 блок-схем, 2 таблицы, количество страниц – 79.

ГЛАВА 1. ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Основные термины

Интернет — всемирная система объединённых *компьютерных сетей* для хранения и передачи информации. Часто упоминается как Всемирная сеть и Глобальная сеть, а также просто Сеть. Построена на базе *стека протоколов TCP/IP***.** На основе Интернета работает *Всемирная паутина* (World Wide Web, WWW) и множество других *систем передачи данных*.

К 30 июня 2012 года число пользователей, регулярно использующих Интернет, составило более чем 2,4 млрд человек, более трети населения Земли пользовалось услугами Интернета [3].

Информационная система — система обработки информации, работающая совместно с организационными ресурсами, такими как люди, технические средства и финансовые ресурсы, которые обеспечивают и распределяют информацию

Интернет-ресурс – *информационная система*, использующая web-технологии на уровне представления и передачи данных, предназначенная для оказания публичных информационных услуг в сети Интернет. Подразумевается, что web-ресурс имеет постоянный адрес (URL) во всемирной сети Интернет.

Информационные технологии— широкий класс дисциплин и областей деятельности, относящихся к *технологиям* создания, сохранения, управления и *обработки данных*, в том числе с применением вычислительной техники. В последнее время под информационными технологиями чаще всего понимают *компьютерные технологи**и***. В частности, ИТ имеют дело с использованием компьютеров и *программного обеспечения* для создания, хранения, обработки, ограничения к передаче и получению информации. Специалистов по компьютерной технике и программированию часто называют ИТ-специалистами. Согласно определению, принятому *ЮНЕСК**О***, ИТ — это

комплекс взаимосвязанных научных, технологических, инженерных дисциплин, изучающих методы эффективной организации труда людей, занятых обработкой и хранением информации; вычислительная техника и методы организации и взаимодействия с людьми и производственным оборудованием, их практические приложения, а также связанные со всем этим социальные, экономические и культурные проблемы. Сами ИТ требуют сложной подготовки, больших первоначальных затрат и наукоемкой техники. Их внедрение должно начинаться с создания математического обеспечения, моделирования, формирования информационных хранилищ для промежуточных данных и решений [4,5,6,7].

Отрасль информационных технологий занимается созданием, развитием и эксплуатацией *информационных систем*. Информационные технологии призваны, основываясь и рационально используя современные достижения в области компьютерной техники и иных высоких технологий, новейших средств коммуникации, программного обеспечения и практического опыта, решать задачи по эффективной организации информационного процесса для снижения затрат времени, труда, энергии и материальных ресурсов во всех сферах человеческой жизни и современного общества. Информационные технологии взаимодействуют и часто составляющей частью входят в сферы услуг, области управления, промышленного производства, социальных процессов. [5]

1.2 Процесс анализа и проектирования будущей системы

Этап проектирования системы начинается с планирования. Для этого приказом по каталогу создается рабочая группа, которая разрабатывает будущую структуру информационной системы. Эта группа должна состоять из наиболее творческих представителей дизайнеров, программистов, креативщиков. Рабочая группа действует постоянно, т.е. будет продолжать

свою работу и после создания первой версии сайта.

Цель: В первую очередь рабочей группе следует четко сформулировать цель создания информационной системы. Вы должны точно знать, куда вы движетесь и чего хотите достигнуть. Если вы знаете конечную цель, менее важные вопросы будут решаться автоматически. Некоторые пожелания:

Сосредоточьтесь на нуждах людей – заказчиках. Вся наша работа: продукция и услуги - предназначена прежде всего для них.

Будьте реалистами: цели должны быть честным и достижимыми. Не обещайте того, чего не можете предоставить своим заказчикам.

Пусть цели вдохновляют не только разработчиков сайта, но и всех сотрудников компании.

Целевая аудитория: С точки зрения маркетинга сайт — это набор информационных блоков и инструментов для взаимодействия с целевой аудиторией, это реальные и потенциальные клиенты и партнеры в одном месте. Поэтому так важно четко определить целевую аудиторию. После этого будет гораздо легче разработать структуру сайта.

Целевая аудитория сайта — это группа интернет-пользователей, на которую сфокусировано содержание сайта; круг посетителей, заинтересованных в информации или услугах, представленных на сайте. Надо знать, как минимум, следующие параметры целевой аудитории:

1. *Объем целевой аудитории* - количество пользователей, которых можно потенциально привлечь на сайт.

2. *Социально-психологические свойства аудитории* - возраст, пол, социальный статус, интеллектуальный уровень, потенциальный круг интересов, специфические свойства ключевой аудитории (консерватизм, следование обычаям, традициям и т.п.). Обычно этими характеристиками можно ограничиться.

3. *Ареал аудитории* - веб-сайты в Интернете, которые могут посещаться представителями целевой аудитории. Ареалы могут определяться как для

целевой аудитории в целом, так и для ее отдельных групп. [6]

Требования к сайту: Изучение практики создания сайтов позволяет сформулировать ряд общих требований, которым должна отвечать информационная система по разработке мультимедийных программных ресурсов. К ним относятся:

- привлекательность содержания;
- простота навигации;
- оперативность обновления информации
- доступность для пользователей;
- единство дизайна всех разделов.

1.3 Разработка веб-приложения

Веб-приложение — клиент-серверное приложение, в котором клиентом выступает браузер, а сервером — веб-сервер. Логика веб-приложения распределена между сервером и клиентом, хранение данных осуществляется, преимущественно, на сервере, обмен информацией происходит по сети. Одним из преимуществ такого подхода является тот факт, что клиенты не зависят от конкретной операционной системы пользователя, поэтому веб-приложения являются межплатформенными сервисами.

Веб-приложения стали широко популярными в конце *1990-х* — начале 2000-х годов.

Технические особенности: Существенное преимущество построения Web приложений для поддержки стандартных функций браузера заключается в том, что функции должны выполняться независимо от операционной системы данного клиента. Вместо того чтобы писать различные версии для *Microsoft Windows*, *Mac OS X*, *GNU/Linux* и других операционных систем, приложение создается один раз для произвольно выбранной платформы и на ней разворачивается. Однако различная реализация *HTML*, *CSS*, *DOM* и

других спецификаций в браузерах может вызвать проблемы при разработке веб-приложений и последующей поддержке. Кроме того, возможность пользователя настраивать многие параметры браузера (например, размер шрифта, цвета, отключение поддержки сценариев) может препятствовать корректной работе приложения.

Другой (менее универсальный) подход заключается в использовании *Adobe Flash*, *Silverlight* или *Java-апплетов* для полной или частичной реализации пользовательского интерфейса. Поскольку большинство браузеров поддерживает эти технологии (как правило, с помощью *плагинов*), Flash- или Java-приложения могут выполняться с легкостью. Так как они предоставляют программисту больший контроль над интерфейсом, они способны обходить многие несовместимости в конфигурациях браузеров, хотя несовместимость между Java или Flash реализациями на стороне клиента может приводить к различным осложнениям.

В связи с архитектурным сходством с традиционными клиент-серверными приложениями, в некотором роде «толстыми» клиентами, существуют споры относительно корректности отнесения подобных систем к веб-приложениям; альтернативный термин «*Богатое Интернет приложение*» (***англ.*** *Rich Internet Applications*).

Устройство веб-приложений: Веб-приложение состоит из клиентской и серверной частей, тем самым реализуя технологию «клиент-сервер».

Клиентская часть реализует пользовательский интерфейс, формирует запросы к серверу и обрабатывает ответы от него.

Серверная часть получает запрос от клиента, выполняет вычисления, после этого формирует веб-страницу и отправляет её клиенту по сети с использованием протокола HTTP [7].

Само веб-приложение может выступать в качестве клиента других служб, например, базы данных или другого веб-приложения, расположенного на другом сервере. Ярким примером веб-приложения является система

управления содержимым статей Википедии: множество её участников могут принимать участие в создании сетевой энциклопедии, используя для этого браузеры своих операционных систем (будь то Microsoft Windows, GNU/Linux или любая другая операционная система) и не загружая дополнительных исполняемых модулей для работы с базой данных статей.

В настоящее время набирает популярность новый подход к разработке веб-приложений, называемый Ajax. При использовании Ajax страницы веб-приложения не перезагружаются целиком, а лишь догружают необходимые данные с сервера, что делает их более интерактивными и производительными. [4]

Для создания веб-приложений на стороне сервера используются разнообразные технологии и любые языки программирования, способные осуществлять вывод в стандартную консоль.

Название	Лицензия	Веб-сервер
ASP	проприетарная	специализированный
ASP.NET	проприетарная	специализированный
C/C++	свободная	практически любой
Java	свободная	множество, в том числе свободных
Perl	свободная	практически любой
PHP	свободная	практически любой
Python	свободная	практически любой
Ruby	свободная	практически любой
Nodejs	MIT License	собственный

На стороне клиента используется:

- Для реализации GUI
    - HTML, XHTML
    - CSS

- Для формирования и обработки запросов, создания интерактивного и независимого от браузера интерфейса:
  - ActiveX
  - Adobe Flash, Adobe Flex
  - Java
  - JavaScript
  - Silverlight

1.4 Анализ имеющихся аналогов WEB-структур для каталогов мультимедийных программных ресурсов, их состав и функции

Информатизация общества вызвана, с одной стороны, необходимостью использования больших объемов информации во всех сферах человеческой деятельности, а с другой стороны, невозможностью формирования и обработки информации традиционными технологиями, средствами связи.

За последние годы роль ПК и ИТ в жизни общества значительно выросла. Человек, эффективно владеющий технологиями, имеет принципиально новый стиль мышления. Невозможно представить современные компании без информационных технологий. Поэтому очень важен перед на качественно новый уровень в использовании информационных технологий.

Каталог мультимедийных программных ресурсов – это комплекс средств интерактивного взаимодействия заказчиков и поставщиков, это качественное обеспечение заказчиков необходимыми в их деле услугами. Каталог мультимедийных программных ресурсов предназначен для улучшения средств связи между заказчиком и организацией, а так же для оперативного обеспечения заказчика информацией.

Функциональные требования к информационной системе, которые описываются, в том числе, и с помощью моделей процессов и структур

данных, являются только частью общих требований, которые содержатся в техническом задании. Раздел требований технического задания к информационной системе содержит следующие подразделы:

- требования к функциональным характеристикам;
- требования к надежности;
- требования к актуальности;
- требования к новизне;
- настраиваемость;
- условия эксплуатации;
- требования к составу и параметрам технических средств.

В разделе «Требования к функциональным характеристикам» должны быть указаны требования к составу выполняемых функций, организации входных и выходных данных. Если предполагается использовать структурное программирование, то и на этапе анализа следует использовать структурный подход, а в случае использования объектно-ориентированных языков разработки - объектный анализ и объектное проектирование. В конкретных случаях оба эти подхода могут использоваться одновременно.

В разделе «Требования к надежности» должны быть четко определены требования к надежности разрабатываемого продукта, а именно, обеспечению надежного функционирования. Речь идет о контроле входной и выходной информации, время и механизмы восстановления работоспособности системы после программных или аппаратных сбоев. Также в этом разделе описывается организация системы безопасности, включая подсистемы контроля доступа, шифрования и т.п.

В разделе «Настраиваемость» определяются требования к адаптационным возможностям ПО, то есть указывается, какие изменения в методах управления и бизнес процессах должны быть предусмотрены.

В разделе «Условия эксплуатации» описывается необходимое обслуживание, которое требуется для полнофункциональной работы

системы. В их число входит создание резервных копий, реиндексирование баз, и, что не менее важно, требования к квалификации персонала.

В разделе «требования к составу и параметрам технических средств» определяется необходимый состав технических средств с указанием их основных технических характеристик.

Динамика изменения требований зависит от выбранной модели жизненного цикла системы, в каскадной модели требования определяются один раз в начале проекта, а в итерационной - уточняются в ходе выполнения проекта. Во втором случае должна быть предусмотрена процедура управления требованиями. Одним из возможных подходов является представление совокупности требований в виде набора атомарных требований - утверждений, между которыми выявляются отношения зависимости.

Модули, обеспечивающие функциональность сайта:

- Администрирование модулей;
- Подключение базы данных для полнофункционального взаимодействия с содержимым сайта, получением всей необходимой информации и т.п.

Структура сайта. В соответствии с предоставляемыми услугами и информацией, которую необходимо изложить на сайте, выберем иерархическую структуру сайта.

Структура сайта. В соответствии с тематикой сайта и разделами, которые необходимо реализовать на сайте, выберем иерархическую структуру сайта. Главная страница сайта является контейнером, в котором отображаются другие страницы. Навигация по сайту должна осуществляться на главной странице посредством использования для этих целей горизонтального меню. [6,7,10]

1.5 Методика разработки WEB-САЙТА

Все методы создания сайтов можно условно разделить на 2 основные группы.

Первая группа методов создания сайтов – это методы ручного написания сайтов на одном или нескольких языках веб-программирования, с использованием как простых, так и визуальных редакторов кода.

В случае статического сайта вполне достаточным для ручного написания будет использование «связки» HTML и CSS, с возможным включением Javascript. Для создания же динамического сайта не обойтись без серверных скриптов, таких как PHP, ASP.NET и т.д.

Создавать и редактировать вручную файлы .php можно даже в обычном «Блокноте», поставляемом с ОС Windows. Для работы с ASP.NET придется дополнительно установить программный продукт Microsoft Visual Studio, который приобретается отдельно.

При использовании «ручных» методов создания сайта дизайн сайта (графическое оформление) также создается вручную. Для этих целей применяются любые графические редакторы по желанию.

Вторая группа методов создания сайтов включает в себя методы автоматизированного создания сайтов: при помощи специальных конструкторов сайтов или же систем управления контентом (CMS).

Конструкторы сайтов – это, как правило, онлайн-системы, позволяющие из готового типового набора модулей и компонентов «собрать» сайт и сразу же разместить его в web. Одни из наиболее популярных конструкторов сайтов – это системы Wordpress, Drupal, Joomla.

Методы создания сайтов с использованием CMS – одни из самых популярных на сегодняшний день. CMS, выражаясь условно, представляет собой некую готовую визуальную и программную оболочку, которую пользователь может заполнить необходимым контентом, а также по своему желанию изменить и настроить.

Автоматизированные методы создания сайтов предусматривают разделение структуры сайта на «дизайн» и «контент». В этом случае легко можно изменять контент, не затрагивая дизайна сайта или его программного кода. При ручном создании сайта разделения структуры сайта на две отдельные «ветви» - дизайн и содержимое - не происходит.

Простые методы:.Методы ручного создания сайтов довольно сложны, ведь они требуют значительных знаний в области веб-программирования или дизайна сайтов. Однако они обладают неоспоримым преимуществом: создавая сайт вручную, всегда можно получить именно то, что хочешь. «Ручные» методы создания сайтов многие «акулы» веб-программирования предпочитают именно поэтому.

Создание сайтов на основе бесплатных онлайн-конструкторов удобно для начинающих веб-мастеров, желающих «испытать свои силы». Преимущественно этот метод подходит для создания небольших простых сайтов, например, сайтов-визиток.

Широкие возможности по созданию сайтов любой сложности предоставляют CMS. Именно этот метод создания сайтов по праву считается одним из наиболее удобных и практичных. Гибкая система настроек, возможность редактирования самой CMS или же отдельных ее элементов, легкость добавления и изменения контента – все это сделало создание сайтов на базе CMS по-настоящему эффективным.

По методу разработки сайты делятся на «статические» и «динамические». «Статические» сайты представляют собой набор размещённых на Интернет-сервере файлов, не содержащий исполняемых на стороне сервера программ.

«Динамические» сайты, в отличии от статических включают в себя набор исполняемых на стороне Интернет-сервера программ, формирующих страницы, просматриваемые посетителем сайта через браузер.

В этом разделе будут рассмотрены особенности, возможности, преимущества

и недостатки «статической» технологии разработки Интернет-сайтов с точки зрения заказчика.

Динамические Интернет-сайты

Итак, опять повторимся — «динамические» Интернет-сайты, в отличии от «статических» сайтов, помимо либо вместо набора размещённых на Интернет-сервере файлов, не содержащих исполняемых на стороне сервера программ, содержат в себе исполняемые на сервере программы, формирующие в ответ на запрос клиентского браузера страницы сайта, демонстрируемые пользователю.

В статьях раздела рассмотрены разные общие вопросы, связанные с этой технологией разработки Интернет-сайтов: особенности, возможности, преимущества и недостатки «динамической» технологии разработки Интернет-сайтов с точки зрения заказчика.

Раздел ни в коей мере не является руководством, справочником либо самоучителем по разработке «динамических» Интернет-сайтов, веб-программированию, а также — администрированию Интернет-сайтов.

По первой главе можно сделать вывод о том, что каталог мультимедийных программных ресурсов очень важен в современных технологиях, в современных система управления взаимоотношения с клиентами ввиду эффективности, простоты, удобства для заказчика, улучшить процесс создания новых информационных систем, облегчить задачу заказчика искать исполнителя, обеспечение всех заказчиков подробной информацией, а также предоставление качественных услуг. [6,7,8]

ГЛАВА 2. ИНФОРМАЦИОННЫЕ МОДЕЛИ И АЛГОРИТМЫ

ФОРМИРОВАНИЯ КАТАЛОГА МУЛЬТИМЕДИЙНЫХ ПРОГРАММНЫХ РЕСУРСОВ

2.1. Реализация WEB-приложения с учетом особенностей современных требований в области информационных технологий

Возможность получить любую информацию своевременно – несомненно, важный фактор устойчивого развития любого каталога мультимедийных программных ресурсов. Возможность знакомиться с жизнью других подобных каталогов, с новым опытом, с новыми технологиями, с новыми проектами и программами позволяет решать возникающие проблемы, находить новых партнеров, выбрать верную стратегию развития.

Каталоги мультимедийных программных ресурсов, как виртуальные источники информации, способны решить проблему способа распространения информации, стоимостью данной информации, недостаточности оперативности и лимит предоставляемой информации, т.к. обеспечивают доступ к размещенной на них информации в любое время, в любом объеме и из любой точки мира.

Первые каталоги мультимедийных программных ресурсов появились в 2000-х годах. Сегодня их каталог насчитывает тысячи сайтов. Каталоги активно используют возможности виртуального способа предоставления информации для представления себя в Интернете, а также чтобы заинтересовать потенциальных клиентов.

Для оценки веб-сайтов были взяты критерии: глубина содержания информации, оперативность ее обновления, стабильность информационного наполнения, доступность страниц для пользователей, простота навигации.

Для оценки глубины содержания были выделены два информационных

блока, обеспечивающих, как нам кажется, реализацию методической и координационной функции каталога мультимедийных программных ресурсов для других каталогов мультимедийных программных ресурсов, фрилансеров и специалистов по информационным технологиям в целом:

- информация о деятельности компании;
- виды предоставляемых услуг и продуктов.

Важное значения для информационного обеспечения развития каталога имеет наличие достойного портфолио. На многих каталогах можно увидеть информацию о созданных ими информационных ресурсах, ссылки на данные ресурсы.

Анализируя каталоги по такому критерию, как «новизна технологий», приходится констатировать, что ни один каталог не содержит разработки программных ресурсов с анимацией, а также с использованием 3D-моделирования.

2.2. Алгоритмы и структуры формирования каталога

Структура сайта – это способ компоновки, расположения, а значит, и подачи информации, который позволяет за короткое время максимально подробно рассказать о предоставляемых организацией услугах. Грамотно выполненная структура, которая подразумевает собой сочетание заголовков, содержания и другой информации, поможет сайту динамично развиваться в Интернете. Структура сайта будет определяться его содержанием, т.е. той информацией, которую вы предполагаете располагать на сайте. Каждый сайт содержит несколько тематических рубрик, соединенных между собой ссылками. Как правило, ссылки на все разделы сайта с краткими анонсами их содержимого приводится на главной странице.

На сайте как месте доступа к информационной среде библиотеки можно выделить следующие блоки:

Информационный: те продукты и услуги, которые каталог предлагает своим пользователям, а также потенциальным заказчикам.

*инструменты для ведения поиска:*Следует стремиться к тому, чтобы каталог полностью отражал состав фонда, и к нему на сайте был организован доступ.

Структура сайта каталога мультимедийных программных ресурсов:

Главная страница - содержит информацию об организации, а также примеры дизайна сайтов;

Дизайн сайтов – здесь представлены примеры дизайна сайтов, которые доступны для заказа

Логотипы – здесь представлены логотипы организаций, для которых услуга создания ресурса была осуществлена

Анимация – примеры дизайна сайтов, использующих анимацию, и доступных для заказа

3D-моделинг – примеры дизайна сайтов, дизайн которых создан на основе 3D-моделирования

Описание блок-схемы разработки сайта:

1) Постановка задачи.
2) Определение требований к сайту.
3) Выявление функциональных возможностей сайта.
4) Разработка структуры и определение связей.
5) Начало – сайт начинает формироваться.
6) Потом все данные вызываются из массива, после чего следует обработка этих данных и разделение их по категориям.
7) Эти данные записываются в БД сайта, а так же в полнотекстовую БД сайта, после чего они выводятся на экран.

Блок-схема разработки сайта:

Рис. 1. Блок-схема разработки сайта.

Функциональная схема:

Рис. 2. Функциональная схема.

В функциональной схеме представлено:

1) Функция ввода, а именно ввод информации для пополнения базы данных и информации для поиска;

2) Поиск информации - обращается к электронному каталогу, который, в свою очередь, обращается к базам данных;

3) При поиске необходимой информации, выводим ее на дисплей.

Структура базы данных:

Рис. 3. Структура базы данных.

БД сайта состоит из электронного каталога, электронный каталог состоит из полнотекстовой базы данных, которая содержит в себе все услуги, а именно:

- Дизайн сайтов

- Логотипы
- Анимация
- 3D-моделинг

Организационная структура:

Рис. 4. Организационная структура.

Главой является учредитель, т.е. лицо, которое проявило инициативу и создало данную организацию для оказания потенциальным заказчикам качественных услуг. Затем идет генеральный директор, который управляет всем предприятием, а также дает поручения администратору (чтобы он поддерживал работоспособность БД), создателю ПБД (чтобы он устранял возможные ошибки сразу после из возникновения), а также делит свою работу по управлению с менеджером, который является ответственным лицом, который отвечает за потенциальных клиентов, заключает с ними

договора, проводит переговоры, а также способствует своевременному обеспечению заказчикам требуемой им информацией и помощью.

Блок-схема взаимодействия потенциального заказчика с системой

Рис. 5. Блок-схема взаимодействия потенциального заказчика с системой

2.3 Языки и программы разработки web-сайта

Новизна данного сайта заключается в том, что позволяет создавать информационные ресурсы, т.е. веб-сайты, используя анимацию, а также 3D-моделирование. Для реализации сайта были использованы ряд программ и

методов, такие как: HTML, JavaScript, MySQL, PHP.

Язык разметки HTML.

HTML (HyperText Markup Language) — это язык разметки

документа, описывающий форму отображения информации на экране компьютера.

При создании документа часто приходится выделять какую-либо часть текста полужирным шрифтом, изменять размер или цвет шрифта, выравнивать текст по центру страницы и т. д. В текстовом редакторе для этого достаточно вынему форматирование. Чтобы пометить текст курсивом, нужно выделить его и нажать кнопку Кур-сив. На языке HTML тот же эффект достигается следующей строкой кода:

Текст

Символ указывает, что текст надо выделить, начиная с этого места, а отмечает конец выделенного фрагмента.

 и принято называть тегами. С помощью тегов описывается вся структура документа. Теги выделяются угловыми скобками "<" и ">", между которыми указывается имя тега. Большинство тегов являются парны-) и соответствующий ему закрываюми, так как есть открывающий тег (). Закрывающий тег отличается наличием косой черты ("/") перед его именем. Есть также теги, вообще не имеющие закрывающего тега, например, тег переноса строки
.

Просматривать HTML-документы можно с помощью специальных программ, которые называют Web-браузерами. Web-браузеры отображают документы с форматированием, выполненным на основе исходного кода, описывающего структуру документа. Результат интерпретации HTML-документа, отображаемый в окне Web-браузера, называется Web-страницей. В отличие от HTML-документа Web-страница может содержать не только текст, но и графику, видео, звуковое сопровождение, может реагировать на действия пользователя и т. д. Кроме того, Web-страница

может быть результатом интерпретации сразу нескольких HTML-документов. Документы в формате HTML имеют расширение html или htm. Прежде чем изучать язык HTML, советую установить на компьютер один из редакторов — FCKeditor или tinyMCE. Эти редакторы написаны на языке программирования JavaScript и работают в Web-браузере.

Язык программирования JavaScript.

JavaScript — это язык программирования, позволяющий сделать Web-страницу интерактивной, то есть реагирующей на действия пользователя. Последовательность инструкций (называемая программой, скриптом или сценарием) выполняется интерпретатором, встроенным в обычный Web-браузер. Иными словами, код программы внедряется в HTML-документ и выполняется на стороне клиента. Для выполнения программы даже не нужно перезагружать Web-страницу. Все программы выполняются в результате возникновения какого-то события. Например, перед отправкой данных формы можно проверить их на допустимые значения и, если значения не соответствуют ожидаемым, запретить отправку данных.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>Первая программа</title>
<meta http-equiv="Content-Type" content="text/html; charset=windows-
 1251">
</head>
<body>
<script type="text/javascript">
<!--
document.write("Hello, world");
//-->
```

```
</script>
<noscript>
<p>Ваш Web-браузер не поддерживает JavaScript</p>
</noscript>
</body>
</html>
```

Набираем код в Блокноте и сохраняем в формате HTML, например, под именем test.html. Запускаем Web-браузер и открываем сохраненный файл.

Возможны следующие варианты: в окне Web-браузера отображена надпись "Hello, world" — значит, все нормально; отобразилась надпись "Ваш Web-браузер не поддерживает JavaScript" и Web-браузер задает вопрос "Запустить скрипты?" — значит, в настройках Web-браузера установлен флажок напротив пункта Подтверждать запуск скриптов. Можно либо установить флажок напротив пункта Разрешить запуск сприптов, либо каждый раз отвечать "Да" на этот вопрос; отобразилась надпись "Ваш Web-браузер не поддерживает JavaScript" и Web-браузер не задает никаких вопросов — значит, в настройках Web-браузера установлен флажок напротив пункта «Запретить запуск скриптов». Надо установить флажок напротив пункта «Разрешить запуск сприптов»; в окне Web-браузера нет никаких надписей — значит, допущена опечатка в коде программы. Следует иметь в виду, что в JavaScript регистр имеет важное значение. Строчные и прописные буквы считаются разными. Более того, каждая буква, каждая кавычка имеет значение. Достаточно ошибиться в одной букве, и вся программа работать не будет. Итак, мы столкнулись с первой проблемой при использовании JavaScript — любой пользователь может отключить запуск скриптов в настройках Web-браузера. Но эта проблема не единственная. Разные Web-браузеры могут по-разному выполнять код программы. По этой причине приходится писать персональный код под каждый Web-браузер. Все примеры скриптов в этой книге написаны под

Microsoft Internet Explorer и могут не работать в других Webбраузерах. Это следует помнить.

MySQL- система управления базами данных.

MySQL — это система управления реляционными базами данных.(СУБД) Сервер MySQL позволяет эффективно работать с данными и обеспечивает быстрый доступ к данным одновременно нескольким пользователям. При этом доступ к данным предоставляется только пользователям, имеющим на это право. Что же такое база данных? Реляционная база данных — это совокупность двумерных таблиц, связанных отношениями друг с другом. Каждая таблица содержит совокупность записей. В свою очередь запись — это набор полей, содержащих связанную информацию. Любое поле в базе данных имеет имя и определенный тип. Имя таблицы должно быть уникальным в пределах базы данных. В свою очередь имя поля должно быть уникальным в пределах таблицы. Для выборки записей из базы данных разработан специализированный язык — SQL (Structured Query Language — структурированный язык запросов). С помощью этого языка можно создавать базы данных и таблицы, добавлять, изменять и удалять данные, получать данные по запросу. Но прежде чем изучать SQL, рассмотрим создание реляционных баз данных.

Язык программирования PHP

PHP — *скриптовый язык* программирования общего назначения, интенсивно применяемый для разработки веб-приложений. В настоящее время поддерживается подавляющим большинством хостинг-провайдеров и является одним из лидеров среди языков программирования, применяющихся для созд*ания динамических веб-сайтов.*

Язык и его интерпретатор разрабатываются группой энтузиастов в рам*ках проекта с открытым кодом*. Проект распространяется под собственной лицензией, несовместимой с GNU GPL.

2.4 Область применения

В области программирования для сети Интернет PHP — один из популярных сценарных языков (наряду с JSP, Perl и языками, используемыми в ASP.NET) благодаря своей простоте, скорости выполнения, богатой функциональности, кроссплатформенности и распространению исходных кодов на основе лицензии PHP.

Популярность в области построения веб-сайтов определяется наличием большого набора встроенных средств для разработки веб-приложений[8]. Основные из них:

- Автоматическое извлечение POST и GET-параметров, а также переменных окружения веб-сервера в предопределённые массивы.
- Взаимодействие с большим количеством различных систем управления базами данных (MySQL, MySQLi, SQLite, PostgreSQL, Oracle (OCI8), Oracle, Microsoft SQL Server, Sybase, ODBC, mSQL, IBM DB2, Cloudscape и Apache Derby, Informix, Ovrimos SQL, Lotus Notes, DB++, DBM, dBase, DBX, FrontBase, FilePro, Ingres II, SESAM, Firebird / InterBase, Paradox File Access, MaxDB, Интерфейс PDO);
- Автоматизированная отправка HTTP-заголовков;
- Работа с HTTP-авторизацией;
- Работа с cookies и сессиями;
- Работа с локальными и удалёнными файлами, сокетами;
- Обработка файлов, загружаемых на сервер;
- Работа с XForms.

В настоящее время PHP используется сотнями тысяч разработчиков. Согласно рейтингу корпорации TIOBE, базирующемся на данных поисковых систем, в июне 2013 года PHP находился на 5 месте с*реди языков программирования*. К крупнейшим сайтам, использующим PHP, относятся Facebook, Wikipedia и др.

Входит в LAMP — распространённый набор программного

обеспечения для создания и хостинга веб-сайтов (Linux, Apache, MySQL, PHP). Синтаксис PHP подобен синтаксису языка Си. Некоторые элементы, такие как ассоциативные массивы и цикл foreach, заимствованы из Perl.

Для работы программы не требуется описывать какие-либо переменные, используемые модули и т. п. Любая программа может начинаться непосредственно с оператора PHP.

Простейшая программа Hello world на PHP выглядит следующим образом:

```
<?php
  echo 'Hello, world!';
?>
```

Также возможен более короткий вариант вывода строки:

```
<?= 'Hello, world!' ?>
```

Открывающий тег вида <?= используется для сокращённой записи конструкций используемых для вывода строки.

PHP исполняет код, находящийся внутри ограничителей, таких как <?php ?>. Всё, что находится вне ограничителей, выводится без изменений. В основном это используется для вставки PHP-кода в HTML-документ, например, так:

```
<html>
 <head>
  <title>Тестируем PHP</title>
 </head>
 <body>
  <?php echo 'Hello, world!'; ?>
 </body>
</html>
```

Помимо ограничителей <?php ?>, допускается использование дополнительных вариантов, таких как <? ?> и <script language="php"> </script>. Кроме того, до версии 6.0 допускается использование

ограничителей языка программирования ASP <% %> (конструкции <? ?> и <% %> могут быть выключены в конфигурационном файле php.ini).

Имена переменных начинаются с символа $, тип переменной объявлять не нужно. Имена переменных, функций и классов чувствительны к регистру. Константы также чувствительны к регистру. Переменные обрабатываются в строках, заключённых в двойные кавычки, и heredoc-строках (строках, созданных при помощи оператора <<<). Переменные в строках, заключенных в одинарные кавычки, не обрабатываются.

PHP рассматривает переход на новую строку как пробел, так же как HTML и другие языки со свободным форматом. Инструкции разделяются с помощью точки с запятой (;), за исключением некоторых случаев, после объявления конструкции if/else и циклов.

Переменные в функцию можно передавать как по значению, так и по ссылке (используется знак &). PHP поддерживает три типа комментариев: в стиле языка Си (ограниченные /* */), C++ (начинающиеся с // и идущие до конца строки) и оболочки UNIX (с # до конца строки).

Типы данных

PHP является языком программирования с динамической типизацией, не требующим указания типа при объявлении переменных, равно как и самого объявления переменных. Преобразования между скалярными типами зачастую осуществляются неявно без дополнительных усилий (впрочем, PHP предоставляет широкие возможности и для явного преобразования типов).

К скалярным типам данных относятся:

Целый тип (integer), вещественный тип данных (float, double), логический тип (boolean), строковый тип (string), и специальный тип NULL.

К нескалярным типам относятся:

«ресурс» (resource), массив (array), объект (object).

К псевдотипам относятся:

Mixed один или несколько необязательных параметров, number число (integer либо float), callback (string или анонимная функция), void отсутствие параметров.

Диапазон целых чисел (integer) в PHP зависит от платформы (обычно, это диапазон 32-битных знаковых целых чисел, то есть, от −2 147 483 648 до 2 147 483 647). Числа можно задавать в десятичной, восьмеричной и шестнадцатеричной системах счисления. Диапазон вещественных чисел (double) также зависит от платформы (для 32-битной архитектуры диапазон позволяет оперировать числами от ±1.7×10−308 до ±1.7×10+308).

PHP предоставляет разработчикам логический тип (boolean), способный принимать только два значения TRUE («истина») и FALSE («ложь»). При преобразовании в логический тип число 0, пустая строка, ноль в строке «0», NULL и пустой массив считаются равными FALSE. Все остальные значения автоматически преобразуются в TRUE.

Специальный тип NULL предназначен для переменных без определённого значения. Единственным значением данного типа является константа NULL. Тип NULL принимают неинициализированные переменные, переменные инициализированные константой NULL, а также переменные, удалённые при помощи конструкции unset().

Ссылки на внешние ресурсы имеют тип «ресурс» (resource). Переменные данного типа, как правило, представляют собой дескриптор, позволяющий управлять внешними объектами, такими как файлы, динамические изображения, результирующие таблицы базы данных и т. п.

Массивы (array) поддерживают числовые и строковые ключи и являются гетерогенными. Массивы могут содержать значения любых типов, включая другие массивы. Порядок элементов и их ключей сохраняется. Не совсем корректно называть php-массивы массивами, на самом деле это, скорее всего, упорядоченный хеш. Возможно неожиданное поведение при

использовании цикла for со счетчиком вместо foreach. Так, например, при сортировке массива с численными индексами функциями из стандартной библиотеки, сортируются и ключи тоже.

Указатель на функцию в PHP может быть представлен замыканием или псевдотипом callback. Замыкание доступно с версии 5.3 и в коде выглядит как простое определение функции, в которую явно можно утянуть значения из контекста, например:

function($args..$argsN) use($ctxVar,$ctxVar1) { definition ; }

callback тип может быть представлен:

Строкой (интерпретируется как название функции);

Массивом где нулевой и первый элемент — строки (интерпретируется как Название статической функции класса); Массивом где нулевой элемент — объект, а первый — строка (интерпретируется как метод у объекта).

Для проверки является ли значение вызываемым следует использовать is_callable($var) обращение к переменным и функциям[править исходный текст].

Обращение к переменным осуществляется с помощью символа $, за которым следует имя переменной. Данная конструкция может быть применена также для создания динами*ческих переменных и функций.* Например:

```
$a = 'I am a'; // Запись значения в переменную $a
echo $a; // Вывод переменной $a
 $b = 'a';
echo $$b; // Вывод переменной $a (дополнительный $ перед переменной $b)
 echo ${'a'}; // Вывод переменной $a
 function_name(); // Вызов функции function_name
$c = 'function_name';
$c(); // Вызов функции function_name
 $d = 'Class_name';
```

$obj = new Class_name; // Создание объекта класса Class_name

$obj = new $d(); // Создание объекта класса Class_name

$obj->b; // Обращение к полю b объекта

$obj->c(); // Вызов метода c() объекта

$obj->$b; // Обращение к полю a объекта, так как $b = 'a'

$obj->$c(); // Вызов метода function_name() объекта, так как $c = 'function_name'

В PHP echo и *print не являются функциями* (хотя print имеет возвращаемое значение), а являются синтаксическими единицами. При их использовании можно опустить скобки.

В данной главе показана новизна работы, решенные в ней задачи, а именно:

- Анализ состояния развития каталогов мультимедийных программных ресурсов и их структура;
- Разработка структуры, алгоритмов и информационных моделей каталога мультимедийных ресурсов;

Так же был представлены все методы создания сайта, использованные программы и схемы.

ГЛАВА 3. РАЗРАБОТКА АЛГОРИТМА ПРОГРАММЫ «GROWTO»

3.1. Руководство пользователя

Сайт создан для каталога мультимедийных программных ресурсов. для корректной работы сайта необходимо выполнить следующие требования:

1) Требования к надежности;
2) Требования к эргономике и технической эстетике;
3) Требования к программному обеспечению.

Требования к надежности

Система должна сохранять работоспособность и обеспечивать восстановление своих функций при возникновении следующих внештатных ситуаций:

– При сбоях в системе электроснабжения аппаратной части, приводящих к перезагрузке ОС, восстановление программы должно происходить после перезапуска ОС и запуска исполняемого файла системы.
– При ошибках в работе аппаратных средств (кроме носителей данных и программ) восстановление функции системы возлагается на ОС.
– При ошибках, связанных с программным обеспечением (ОС и драйверы устройств), восстановление работоспособности возлагается на ОС.

Для защиты аппаратуры от бросков напряжения и коммутационных помех должны применяться сетевые фильтры.

Требования к эргономике и технической эстетике

Взаимодействие пользователей с прикладным программным обеспечением, входящим в состав системы должно осуществляться посредством визуального графического интерфейса (GUI). Интерфейс системы должен быть понятным и удобным, не должен быть перегружен графическими элементами и должен обеспечивать быстрое отображение экранных форм. Навигационные элементы должны быть выполнены в удобной для пользователя форме. Средства редактирования информации должны удовлетворять принятым соглашениям в части использования функциональных клавиш, режимов работы, поиска, использования оконной системы. Ввод-вывод данных системы, прием управляющих команд и отображение результатов их исполнения должны выполняться в интерактивном режиме. Интерфейс должен соответствовать современным

эргономическим требованиям и обеспечивать удобный доступ к основным функциям и операциям системы. Интерфейс должен быть рассчитан на преимущественное использование манипулятора типа «мышь», то есть управление системой должно осуществляться с помощью набора экранных меню, кнопок, значков и т. п. элементов. Клавиатурный режим ввода должен используется главным образом при заполнении и/или редактировании текстовых и числовых полей экранных форм. Все надписи экранных форм, а также сообщения, выдаваемые пользователю (кроме системных сообщений) должны быть на русском языке. Система должна обеспечивать корректную обработку аварийных ситуаций, вызванных неверными действиями пользователей, неверным форматом или недопустимыми значениями входных данных. В указанных случаях система должна выдавать пользователю соответствующие сообщения, после чего возвращаться в рабочее состояние, предшествовавшее неверной (недопустимой) команде или некорректному вводу данных.

Требования к программному обеспечению.

На компьютере, где будет работать сайт, должен существовать следующий пакет программ:

1) Интернет браузер;
2) Microsoft office;
3) Программа для просмотра видео и изображений;
4) Download master для скачивания файлов;
5) Текстовый редактор;
6) Djvu reader.

3.2. Интерфейс веб- сайта.

Главная страница - отображает информацию об организации каталога мультимедийных ресурсов, а также левое меню с описанием услуг по дизайну и служит для фильтрации поиска и верхнее меню с описанием базовых услуг, а именно:

- Дизайн сайтов
- Логотипы
- Анимация
- 3D-моделинг

Рис. 6. Главная страница.

Дизайн сайтов – отображает главные дизайны сайтов для заказа. Левок боковое меню служит для фильтрации поиска по категориям

Рис. 7. Дизайн сайтов.

Анимация – список дизайна сайтов, возможных для заказа, и при этом использующих анимацию.

Рис. 8. Анимация.

3D-моделинг – примеры дизайна сайтов, доступных для заказа и при этом созданных при помощи 3D-моделирования.

Рис. 9. 3D-моделинг

ГЛАВА 4. БЕЗОПАСНОСТЬ ЖИЗНЕДЕЯТЕЛЬНОСТИ

4.1. Гиподинамия и влияние ее на здоровье человека.

Еще в древности было замечено, что физическая активность способствует формированию сильного и выносливого человека, а неподвижность ведет к снижению работоспособности, заболеваниям и тучности. Все это происходит вследствие нарушения обмена веществ. Уменьшение энергетического обмена, связанное с изменением интенсивности распада и окисления органических веществ, приводит к нарушению биосинтеза, а также к изменению кальциевого обмена в организме. Вследствие этого в костях происходят глубокие изменения. Прежде всего, они начинают терять кальций. Это приводит к тому, что кость делается рыхлой, менее прочной. Кальций попадает в кровь, оседает на стенках кровеносных сосудов, они склерозируются, т. е. пропитываются кальцием, теряют эластичность и делаются ломкими. Способность крови к свертыванию резко возрастает. Возникает угроза образования кровяных сгустков (тромбов) в сосудах. Содержание большого количества кальция в крови способствует образованию камней в почках.

Отсутствие мышечной нагрузки снижает интенсивность энергетического обмена, что отрицательно сказывается на скелетных и сердечной мышцах. Кроме того, малое количество нервных импульсов, идущих от работающих мышц, снижает тонус нервной системы, утрачиваются приобретенные ранее навыки, не образуются новые. Все это самым отрицательным образом отражается на здоровье. Следует учесть также следующее. Сидячий образ жизни приводит к тому, что хрящ постепенно становится менее эластичным, теряет гибкость. Это может повлечь снижение амплитуды дыхательных движений и потерю гибкости тела. Но особенно сильно от неподвижности или малой подвижности страдают суставы.

Характер движения в суставе определен его строением. В коленном суставе ногу можно только сгибать и разгибать, а в тазобедренном суставе

движения могут совершаться во всех направлениях. Однако амплитуда движений зависит от тренировки. При недостаточной подвижности связки теряют эластичность. В полость сустава при движении выделяется недостаточное количество суставной жидкости, играющей роль смазки. Все это затрудняет работу сустава.

Недостаточная нагрузка влияет и на кровообращение в суставе. В результате питание костной ткани нарушается, формирование суставного хряща, покрывающего головку и суставную впадину сочленяющихся костей, да и самой кости идет неправильно, что приводит к различным заболеваниям. Но дело не ограничивается только этим. Нарушение кровообращения может привести к неравномерному росту костной ткани, вследствие чего возникает разрыхление одних участков и уплотнение других. Форма костей в результате этого может стать неправильной, а сустав потерять подвижность.

Рис. 10. Гиподинамия и влияние ее на здоровье человека.

Гиподинамия – это слабость мышечных тканей, возникающая из-за крайне малой двигательной активности. Современному человеку доступны все блага цивилизации: автомобили, магазины на каждом шагу, сидячая работа, интернет. Все это, конечно, хорошо, но проблема в том, что для человеческого организма малоподвижный образ жизни – смерти подобно.

Ведь самой природой заложено, что мы должны много и активно двигаться.

С другой стороны, не стоит думать, что если делать каждый день по 50 отжиманий или по 100 раз качать пресс, то для тела этого будет достаточно. Дело в том, что когда мышцы постоянно работают в одном и том же режиме, ежедневно выполняют одинаковые действия (допустим, вы каждый день поднимаетесь на 12 этаж пешком), то такая ограниченность движений тоже в итоге приведет к гиподинамии.

Как распознать гиподинамию?

1. Если ваши мышцы недостаточно часто сокращаются, то, по задумке природы, «ненужные» органы атрофируются. Конечно, на это требуется время, поэтому, как только вы заметили, что простые действия (например, пройти пешком два квартала), вызывают у вас одышку и боли в ногах, нужно бить тревогу – гиподинамия рядом! Ведь гиподинамия и здоровье человека неразрывно связаны.

2. Если ваш вес непрерывно растет, это значит, что организм не получает необходимой физической активности. А калории, запасаемые им для мышечной нагрузки, вместо этого превращаются в жир. При этом замедляется обмен веществ, и формы «расползаются» еще быстрее.

3. Вас постоянно тянет к холодильнику, хотя, казалось бы, ужин был не так давно? Вы удивитесь, но это тоже косвенный признак гиподинамии. Дело в том, что когда человек достаточно много двигается, то жиры расщепляются и попадают в кровь, поддерживая в ней нужный уровень сахара. Поэтому вы не будете чувствовать голода, а съесть захочется ровно столько, сколько нужно организму для нормальной жизнедеятельности. Если же движения мало, то уровень сахара быстро падает, в результате человек слабеет и пытается компенсировать недостаток сил с помощью поглощения жирных и сладких продуктов.

Гиподинамия и ее последствия

В течение всей жизни на человека оказывают влияние самые

разнообразные факторы внешней и внутренней среды. Их огромное количество. Однако, несмотря на большое количество, все эти факторы можно ранжировать в порядке их значимости. Для здоровья. Это сделала Всемирная Организация Здравоохранения. Из 200 основных выделенных факторов, которые оказывают самое значительное влияние на человека, первые четыре места занимают гиподинамия (недостаток движения), неправильное питание (и, прежде всего, избыточный вес), вредные привычки (потребление алкоголя, наркотиков и других веществ) и неблагоприятная экологическая обстановка.

Существующая система образования не только не способствует улучшению здоровья учащихся, но и зачастую требует огромного количества движений для своего развития, не менее 50-60% времени в режиме дня должно отводиться двигательной активности.

Вместе с тем, потребность в движении у учащейся молодежи самостоятельными движениями удовлетворяется только на 8-20%.

Многочисленные исследования показывают, что существующая система физического воспитания и программа не способствуют гармоничному развитию детей и подростков и требуют совершенствования, новых решений, оптимального воздействия всех форм, средств и методов с целью сохранения и укрепления здоровья учащихся.

К заболеваниям, связанным с гипокинезией, относятся сердечно-сосудистые, нервные, желудочно-кишечные расстройства, костные, мышечные и хрящевые изменения и др.

Как же бороться с гиподинамией?

Как видите, влияние гиподинамии на человека достаточно сильно и пагубно для него. Но бороться с ней можно и нужно. Главные враги гиподинамии – регулярные и разнообразные физические нагрузки. О том, что необходимы каждодневная зарядка и ходьба пешком, вы, наверное, уже

догадались. Но есть еще одно эффективное средство против этого недуга – изометрическая гимнастика (ее еще называют «карманной»). Эти упражнения удобны тем, что почти незаметны посторонним, а потому их можно делать где угодно. Кроме того, они основаны на сильном напряжении мышц, и повторяя их всего раз в день, можно быть уверенной, что необходимая мышечная нагрузка получена. Итак, вот они:

- Вытяните руки, упритесь полусогнутыми пальцами в поверхность стола. Сильно вдохнув, на выдохе бережно, но сильно вдавите пальцы в стол. Давить нужно примерно 5-6 секунд, после чего расслабиться. Отдохнув 30 секунд, снова проделайте упражнение.

- Подсуньте руки под стол и тыльной стороной ладоней с силой толкайте крышку стола вверх. Толкать нужно 5-6 секунд, через полминуты повторить.

- Сцепив ладони сзади на шее, старайтесь нагнуть ее вперед, при этом сопротивляясь всеми мышцами шеи. «Боритесь» 10 секунд, через 30 секунд повторите.

- Сядьте на стул, обхватите ногами его ножки и, напрягаясь, сожмите ноги как можно сильнее. Сжимайте по 10 секунд через каждые полминуты.

- Сцепите кисти вытянутых рук в замок, и, не сгибая рук, пытайтесь разомкнуть их. После полуминутного отдыха повторите.

Как видите, профилактика гиподинамии довольно проста, и, соблюдая эти несложные рекомендации, вы очень скоро сможете попрощаться с эти недугом.

4.2. Организация рабочего места, оснащенного компьютером

Научно-технический прогресс внес серьезные изменения в условия производственной деятельности работников умственного труда. Их труд стал более интенсивным, напряженным, требующим значительных затрат

умственной, эмоциональной и физической энергии. Это потребовало комплексного решения проблем эргономики, гигиены и организации труда, регламентации режимов труда и отдыха.

В настоящее время компьютерная техника широко применяется во всех областях деятельности человека. При работе с компьютером человек подвергается воздействию ряда опасных и вредных производственных факторов: электромагнитных полей (диапазон радиочастот: ВЧ, УВЧ и СВЧ), инфракрасного и ионизирующего излучений, шума и вибрации, статического электричества и др.

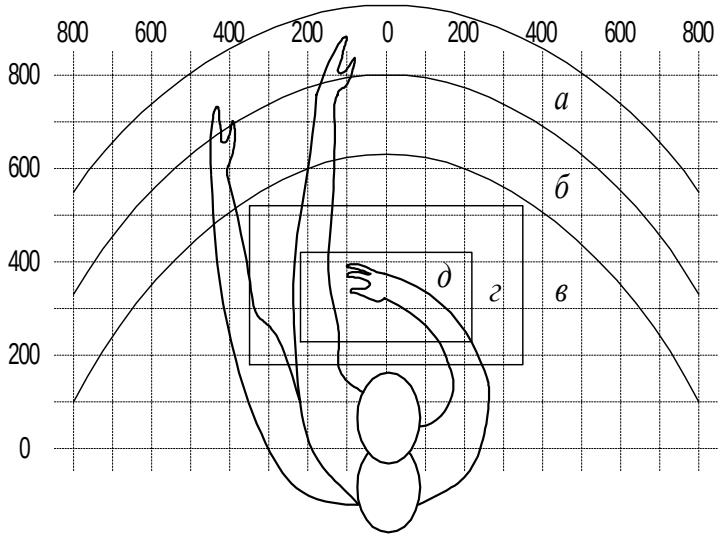
Работа с компьютером характеризуется значительным умственным напряжением и нервно-эмоциональной нагрузкой операторов, высокой напряженностью зрительной работы и достаточно большой нагрузкой на мышцы рук при работе с клавиатурой ЭВМ. Большое значение имеет рациональная конструкция и расположение элементов рабочего места, что важно для поддержания оптимальной рабочей позы человека-оператора.

В процессе работы с компьютером необходимо соблюдать правильный режим труда и отдыха. В противном случае у персонала отмечаются значительное напряжение зрительного аппарата с появлением жалоб на неудовлетворенность работой, головные боли, раздражительность, нарушение сна, усталость и болезненные ощущения в глазах, в пояснице, в области шеи и руках.

Большое значение имеет также характер работы. В частности, при организации рабочего места программиста должны быть соблюдены следующие основные условия: оптимальное размещение оборудования, входящего в состав рабочего места и достаточное рабочее пространство, позволяющее осуществлять все необходимые движения и перемещения.

Главными элементами рабочего места программиста являются стол и кресло. Основным рабочим положением является положение сидя.

Рабочая поза сидя вызывает минимальное утомление программиста. Рациональная планировка рабочего места предусматривает четкий порядок и постоянство размещения предметов, средств труда и документации. То, что требуется для выполнения работ чаще, расположено в зоне легкой досягаемости рабочего пространства.



а - зона максимальной досягаемости;
б - зона досягаемости пальцев при вытянутой руке;
в - зона легкой досягаемости ладони;
г - оптимальное пространство для грубой ручной работы;
д - оптимальное пространство для тонкой ручной работы.

Рис.11. Зоны досягаемости рук в горизонтальной

При оборудовании рабочего места необходимо установить монитор на специальном столике так, чтобы задняя панель была обращена к стене (так как около нее зарегистрирован максимальный уровень напряженности электрического поля), экран не должен располагаться напротив окна или других прямых источников света, дающих блики на экране.

Рис.12. Рекомендуемое положение во время работы за компьютером

Стол, на котором устанавливается монитор, должен быть достаточной длины, чтобы расстояние до экрана составляло 60-70 (не ближе 50) см, и в то же время можно было работать с клавиатурой в непосредственной близости от пользователя (30-40 см). Конструкция рабочей мебели (столы, кресла, стулья) должна обеспечивать возможность индивидуальной регулировки соответственно росту работающего и создавать удобную позу. Часто используемые предметы труда должны находится в оптимальной рабочей зоне, на одном расстоянии от глаз работающего. На поверхности рабочего стола необходимо разместить подставку для документов, расстояние которой от глаз должно быть аналогичным расстоянию от глаз до клавиатуры. Рабочее кресло должно иметь подлокотники. На рабочем месте необходимо предусмотреть подставку для ног.

Для того чтобы устранить блики на экране, монитор должен быть установлен перпендикулярно столу, а пользователь должен смотреть на экран несколько сверху вниз (10° от горизонтальной линии). Условия освещенности в комнате играют большую роль в сохранении зрительного комфорта. С одной стороны, ничто не должно мешать восприятию информации с экрана, с другой - пользователь должен хорошо видеть клавиатуру, бумажные тексты, которыми приходится пользоваться, а также общую обстановку и людей, с которыми приходится общаться при работе.

Рис.13. Удобное рабочее место с "Г-образным" столом

Общая освещенность в комнате не должна быть слишком большой, но и не слишком малой, она должна быть в пределах 300-500 люкс. Если помещение светлое, то окна должны иметь шторы или жалюзи. Рабочие места пользователей дисплеев желательно не располагать непосредственно у окон. Во всех случаях экран монитора следует ориентировать так, чтобы он не давал бликов, а именно - под углом к окну, близким к прямому. Искусственное освещение не должно быть слишком ярким. Но помимо общих ламп, освещающих комнату, необходима местная яркая (не менее 60 Вт) лампа с хорошим плотным абажуром, освещающая только текст, с которым работает пользователь. Она должна иметь возможность ориентации в разных направлениях и быть оснащена устройством для регулирования яркости. Лампы накаливания предпочтительнее люминесцентных, т.к. последние дают пульсирующий свет, в определенных условиях усиливающий мерцание экрана дисплея.

А) Линия взора параллельна окну(рекомендуется)

Б) Яркий свет в поле зрения(не рекомендуется)

Рис.14. Расположение монитора относительно окна

А) Отражение света лампы от поверхности стола и клавиатуры (не рекомендуется)

Б) Блик от лампы на экране монитора (не рекомендуется)

Рис.15. Расположение источника искусственного освещения относительно монитора

Рис.16. Правильное расположение монитора относительно стены и источника света

Перед началом работы с монитором необходимо установить с помощью рукояток наиболее комфортные контрастность и яркость на экране. Они подбираются индивидуально, так как слишком низкая контрастность и высокая яркость могут приводить к быстрому утомлению.

При подборе светового режима на рабочем месте пользователя дисплея необходимо учитывать то, что у лиц после 40 лет возникают возрастные изменения в зрительной системе (сужение зрачка, пожелтение хрусталика, снижение зрительной активности и контрастной чувствительности сетчатки). Все это требует усиления яркости экрана и дополнительной освещенности рабочего места (бумажного текста). У молодых лиц при зрительно-напряженной работе наибольшую нагрузку несет аккомодационная система глаза, которая во время работы находится в постоянном напряжении. Это может приводить к астенопическим явлениям, возникновению нарушений в аккомодационной системе глаза .

У взрослых с близорукостью, которые постоянно носят очки, другие очки для работы с компьютером необходимы только в том случае, если в своих очках пользователь с трудом читает газетный шрифт с расстояния 60-70 см (до экрана) и 30-33 см (до печатного текста) от глаз. В случае если с одними и теми же линзами чтение с обоих расстояний невозможно, назначают бифокальные очки.

Заключение

В результате выполнения данной выпускной квалификационной работы были решены следующие задачи:

- проделан анализ существующих аналогов каталога мультимедийных программных ресурсов;
- разработан алгоритм формирования каталога мультимедийных программных ресурсов;
- разработаны функциональная, организационная структуры каталога;
- создана база данных, а также полнотекстовая база данных с электронным каталогом;
- создан каталог мультимедийных программных ресурсов.

В результате анализа существующих каталогов мультимедийных программных ресурсов была выявлена одна из наиболее распространенных ошибок – недостаточность визуализации при проектировании. Эта ошибка была устранена в нашем проекте и, в конечном итоге, разработан полноценный сайт с возможностью параллельной работы клиента и разработчика в режиме реального времени в on-line. В данной работе каталог обращен на доступность, надежность и дизайн информации.

Разработка может быть использована в любой организации, которая предоставляет услуги проектирования и визуализации проектов в режиме удаленного доступа.

СПИСОК ЛИТЕРАТУРЫ

1. И.А.Каримов. Постановление Президента РУз от 21.03.2012 г. № ПП-1730 «О мерах по дальнейшему внедрению и развитию современных информационно-коммуникационных технологий»
2. И.А. Каримов. «Узбекистан должен стать страной с высоким уровнем внедрения ИКТ» выступление на расширенном заседании Кабинета Министров 19.01.2013
3. С. Айзекс «Dynamic html. Секреты создания интерактивных веб-страниц»
4. Д. Альховник. « Уроки по html для начинающих»
5. Николай Прохоренко «Html. Java Script. Php, MySQL»
6. Информационно-поисковый ресурс http://google.ru/
7. Информационно-справочный ресурс http://wikipedia.org/
8. http://w3schools.com/
9. http://metanit.com/
10. Документация по php
11. Документация по drupal

Приложение (программный код)

```
<?php
/**
 * @file
 * The PHP page that serves all page requests on a Drupal installation.
 *
 * The routines here dispatch control to the appropriate handler, which then
 * prints the appropriate page.
 *
 * All Drupal code is released under the GNU General Public License.
 * See COPYRIGHT.txt and LICENSE.txt.
 */
/**
 * Root directory of Drupal installation.
 */
define('DRUPAL_ROOT', getcwd());
require_once DRUPAL_ROOT . '/includes/bootstrap.inc';
drupal_bootstrap(DRUPAL_BOOTSTRAP_FULL);
menu_execute_active_handler();
<?php
/**
 * @file
 * The theme system, which controls the output of Drupal.
 *
 * The theme system allows for nearly all output of the Drupal system to be
```

```
 * customized by user themes.
 */

/**
 * @defgroup content_flags Content markers
 * @{
 * Markers used by theme_mark() and node_mark() to designate content.
 * @see theme_mark(), node_mark()
 */
/**
 * Mark content as read.
 */
define('MARK_READ', 0);
/**
 * Mark content as being new.
 */
define('MARK_NEW', 1);
/**
 * Mark content as being updated.
 */
define('MARK_UPDATED', 2);
/**
 * @} End of "Content markers".
 */
/**
 * Determines if a theme is available to use.
 *
 * @param $theme
 *   Either the name of a theme or a full theme object.
```

```
 *
 * @return
 *   Boolean TRUE if the theme is enabled or is the site administration theme;
 *   FALSE otherwise.
 */
function drupal_theme_access($theme) {
  if (is_object($theme)) {
    return _drupal_theme_access($theme);
  }
  else {
    $themes = list_themes();
    return isset($themes[$theme]) && _drupal_theme_access($themes[$theme]);
  }
}
/**
 * Helper function for determining access to a theme.
 *
 * @see drupal_theme_access()
 */
function _drupal_theme_access($theme) {
  $admin_theme = variable_get('admin_theme');
  return !empty($theme->status) || ($admin_theme && $theme->name ==
$admin_theme);
}
/**
 * Initialize the theme system by loading the theme.
 */
function drupal_theme_initialize() {
  global $theme, $user, $theme_key;
```

```
  // If $theme is already set, assume the others are set, too, and do nothing
  if (isset($theme)) {
    return;
  }
  drupal_bootstrap(DRUPAL_BOOTSTRAP_DATABASE);
  $themes = list_themes();
  // Only select the user selected theme if it is available in the
  // list of themes that can be accessed.
  $theme = !empty($user->theme) && drupal_theme_access($user->theme) ? $user-
>theme : variable_get('theme_default', 'bartik');
  // Allow modules to override the theme. Validation has already been performed
  // inside menu_get_custom_theme(), so we do not need to check it again here.
  $custom_theme = menu_get_custom_theme();
  $theme = !empty($custom_theme) ? $custom_theme : $theme;
  // Store the identifier for retrieving theme settings with.
  $theme_key = $theme;
  // Find all our ancestor themes and put them in an array.
  $base_theme = array();
  $ancestor = $theme;
  while ($ancestor && isset($themes[$ancestor]->base_theme)) {
    $ancestor = $themes[$ancestor]->base_theme;
    $base_theme[] = $themes[$ancestor];
  }
  _drupal_theme_initialize($themes[$theme], array_reverse($base_theme));
  // Themes can have alter functions, so reset the drupal_alter() cache.
  drupal_static_reset('drupal_alter');
  // Provide the page with information about the theme that's used, so that a
  // later Ajax request can be rendered using the same theme.
```

```
  // @see ajax_base_page_theme()
  $setting['ajaxPageState'] = array(
    'theme' => $theme_key,
    'theme_token' => drupal_get_token($theme_key),
  );
  drupal_add_js($setting, 'setting');
}
/**
 * Initialize the theme system given already loaded information. This
 * function is useful to initialize a theme when no database is present.
 *
 * @param $theme
 *   An object with the following information:
 *     filename
 *       The .info file for this theme. The 'path' to
 *       the theme will be in this file's directory. (Required)
 *     owner
 *       The path to the .theme file or the .engine file to load for
 *       the theme. (Required)
 *     stylesheet
 *       The primary stylesheet for the theme. (Optional)
 *     engine
 *       The name of theme engine to use. (Optional)
 * @param $base_theme
 *   An optional array of objects that represent the 'base theme' if the
 *   theme is meant to be derivative of another theme. It requires
 *   the same information as the $theme object. It should be in
 *   'oldest first' order, meaning the top level of the chain will
 *   be first.
```

```
 * @param $registry_callback
 *   The callback to invoke to set the theme registry.
 */
function _drupal_theme_initialize($theme, $base_theme = array(),
$registry_callback = '_theme_load_registry') {
  global $theme_info, $base_theme_info, $theme_engine, $theme_path;
  $theme_info = $theme;
  $base_theme_info = $base_theme;
  $theme_path = dirname($theme->filename);
  // Prepare stylesheets from this theme as well as all ancestor themes.
  // We work it this way so that we can have child themes override parent
  // theme stylesheets easily.
  $final_stylesheets = array();
  // Grab stylesheets from base theme
  foreach ($base_theme as $base) {
    if (!empty($base->stylesheets)) {
      foreach ($base->stylesheets as $media => $stylesheets) {
        foreach ($stylesheets as $name => $stylesheet) {
          $final_stylesheets[$media][$name] = $stylesheet;
        }
      }
    }
  }
  // Add stylesheets used by this theme.
  if (!empty($theme->stylesheets)) {
    foreach ($theme->stylesheets as $media => $stylesheets) {
      foreach ($stylesheets as $name => $stylesheet) {
        $final_stylesheets[$media][$name] = $stylesheet;
      }
```

```
    }
  }
  // And now add the stylesheets properly
  foreach ($final_stylesheets as $media => $stylesheets) {
    foreach ($stylesheets as $stylesheet) {
      drupal_add_css($stylesheet, array('group' => CSS_THEME, 'every_page' =>
TRUE, 'media' => $media));
    }
  }
  // Do basically the same as the above for scripts
  $final_scripts = array();
  // Grab scripts from base theme
  foreach ($base_theme as $base) {
    if (!empty($base->scripts)) {
      foreach ($base->scripts as $name => $script) {
        $final_scripts[$name] = $script;
      }
    }
  }
  // Add scripts used by this theme.
  if (!empty($theme->scripts)) {
    foreach ($theme->scripts as $name => $script) {
      $final_scripts[$name] = $script;
    }
  }
  // Add scripts used by this theme.
  foreach ($final_scripts as $script) {
    drupal_add_js($script, array('group' => JS_THEME, 'every_page' => TRUE));
  }
```

```
$theme_engine = NULL;

// Initialize the theme.
if (isset($theme->engine)) {
  // Include the engine.
  include_once DRUPAL_ROOT . '/' . $theme->owner;
  $theme_engine = $theme->engine;
  if (function_exists($theme_engine . '_init')) {
    foreach ($base_theme as $base) {
      call_user_func($theme_engine . '_init', $base);
    }
    call_user_func($theme_engine . '_init', $theme);
  }
}
else {
  // include non-engine theme files
  foreach ($base_theme as $base) {
    // Include the theme file or the engine.
    if (!empty($base->owner)) {
      include_once DRUPAL_ROOT . '/' . $base->owner;
    }
  }
  // and our theme gets one too.
  if (!empty($theme->owner)) {
    include_once DRUPAL_ROOT . '/' . $theme->owner;
  }
}
if (isset($registry_callback)) {
  _theme_registry_callback($registry_callback, array($theme, $base_theme,
```

```
$theme_engine));
  }
}

/**
 * Get the theme registry.
 *
 * @param $complete
 *   Optional boolean to indicate whether to return the complete theme registry
 *   array or an instance of the ThemeRegistry class. If TRUE, the complete
 *   theme registry array will be returned. This is useful if you want to
 *   foreach over the whole registry, use array_* functions or inspect it in a
 *   debugger. If FALSE, an instance of the ThemeRegistry class will be
 *   returned, this provides an ArrayObject which allows it to be accessed
 *   with array syntax and  isset(), and should be more lightweight
 *   than the full registry. Defaults to TRUE.
 *
 * @return
 *   The complete theme registry array, or an instance of the ThemeRegistry
 *   class.
 */
function theme_get_registry($complete = TRUE) {
  // Use the advanced drupal_static() pattern, since this is called very often.
  static $drupal_static_fast;
  if (!isset($drupal_static_fast)) {
    $drupal_static_fast['registry'] = &drupal_static('theme_get_registry');
  }
  $theme_registry = &$drupal_static_fast['registry'];
```

```
  // Initialize the theme, if this is called early in the bootstrap, or after
  // static variables have been reset.
  if (!is_array($theme_registry)) {
    drupal_theme_initialize();
    $theme_registry = array();
  }
  $key = (int) $complete;
  if (!isset($theme_registry[$key])) {
    list($callback, $arguments) = _theme_registry_callback();
    if (!$complete) {
      $arguments[] = FALSE;
    }
    $theme_registry[$key] = call_user_func_array($callback, $arguments);
  }

  return $theme_registry[$key];
}
/**
 * Set the callback that will be used by theme_get_registry() to fetch the registry.
 *
 * @param $callback
 *   The name of the callback function.
 * @param $arguments
 *   The arguments to pass to the function.
 */
function _theme_registry_callback($callback = NULL, array $arguments = array())
{
  static $stored;
  if (isset($callback)) {
```

```
    $stored = array($callback, $arguments);
  }
  return $stored;
}

/**
 * Get the theme_registry cache; if it doesn't exist, build it.
 *
 * @param $theme
 *   The loaded $theme object as returned by list_themes().
 * @param $base_theme
 *   An array of loaded $theme objects representing the ancestor themes in
 *   oldest first order.
 * @param $theme_engine
 *   The name of the theme engine.
 * @param $complete
 *   Whether to load the complete theme registry or an instance of the
 *   ThemeRegistry class.
 *
 * @return
 *   The theme registry array, or an instance of the ThemeRegistry class.
 */
function _theme_load_registry($theme, $base_theme = NULL, $theme_engine =
NULL, $complete = TRUE) {
  if ($complete) {
    // Check the theme registry cache; if it exists, use it.
    $cached = cache_get("theme_registry:$theme->name");
    if (isset($cached->data)) {
      $registry = $cached->data;
```

```
    }
    else {
      // If not, build one and cache it.
      $registry = _theme_build_registry($theme, $base_theme, $theme_engine);
      // Only persist this registry if all modules are loaded. This assures a
      // complete set of theme hooks.
      if (module_load_all(NULL)) {
        _theme_save_registry($theme, $registry);
      }
    }
    return $registry;
  }
  else {
    return new ThemeRegistry('theme_registry:runtime:' . $theme->name, 'cache');
  }
}

/**
 * Write the theme_registry cache into the database.
 */
function _theme_save_registry($theme, $registry) {
  cache_set("theme_registry:$theme->name", $registry);
}

/**
 * Force the system to rebuild the theme registry; this should be called
 * when modules are added to the system, or when a dynamic system needs
 * to add more theme hooks.
 */
```

```
function drupal_theme_rebuild() {
  drupal_static_reset('theme_get_registry');
  cache_clear_all('theme_registry', 'cache', TRUE);
}

/**
 * Builds the run-time theme registry.
 *
 * Extends DrupalCacheArray to allow the theme registry to be accessed as a
 * complete registry, while internally caching only the parts of the registry
 * that are actually in use on the site. On cache misses the complete
 * theme registry is loaded and used to update the run-time cache.
 */
class ThemeRegistry Extends DrupalCacheArray {

  /**
   * Whether the partial registry can be persisted to the cache.
   *
   * This is only allowed if all modules and the request method is GET. theme()
   * should be very rarely called on POST requests and this avoids polluting
   * the runtime cache.
   */
  protected $persistable;
  /**
   * The complete theme registry array.
   */
  protected $completeRegistry;
  function __construct($cid, $bin) {
    $this->cid = $cid;
```

```
    $this->bin = $bin;
    $this->persistable = module_load_all(NULL) &&
$_SERVER['REQUEST_METHOD'] == 'GET';

    $data = array();
    if ($this->persistable && $cached = cache_get($this->cid, $this->bin)) {
      $data = $cached->data;
    }
    else {
      // If there is no runtime cache stored, fetch the full theme registry,
      // but then initialize each value to NULL. This allows offsetExists()
      // to function correctly on non-registered theme hooks without triggering
      // a call to resolveCacheMiss().
      $data = $this->initializeRegistry();
      if ($this->persistable) {
        $this->set($data);
      }
    }
    $this->storage = $data;
  }

  /**
   * Initializes the full theme registry.
   *
   * @return
   *   An array with the keys of the full theme registry, but the values
   *   initialized to NULL.
   */
  function initializeRegistry() {
```

```
    $this->completeRegistry = theme_get_registry();

    return array_fill_keys(array_keys($this->completeRegistry), NULL);
  }

  public function offsetExists($offset) {
    // Since the theme registry allows for theme hooks to be requested that
    // are not registered, just check the existence of the key in the registry.
    // Use array_key_exists() here since a NULL value indicates that the theme
    // hook exists but has not yet been requested.
    return array_key_exists($offset, $this->storage);
  }

  public function offsetGet($offset) {
    // If the offset is set but empty, it is a registered theme hook that has
    // not yet been requested. Offsets that do not exist at all were not
    // registered in hook_theme().
    if (isset($this->storage[$offset])) {
      return $this->storage[$offset];
    }
    elseif (array_key_exists($offset, $this->storage)) {
      return $this->resolveCacheMiss($offset);
    }
  }

  public function resolveCacheMiss($offset) {
    if (!isset($this->completeRegistry)) {
      $this->completeRegistry = theme_get_registry();
    }
```

```
    $this->storage[$offset] = $this->completeRegistry[$offset];
    if ($this->persistable) {
      $this->persist($offset);
    }
    return $this->storage[$offset];
  }

  public function set($data, $lock = TRUE) {
    $lock_name = $this->cid . ':' . $this->bin;
    if (!$lock || lock_acquire($lock_name)) {
      if ($cached = cache_get($this->cid, $this->bin)) {
        // Use array merge instead of union so that filled in values in $data
        // overwrite empty values in the current cache.
        $data = array_merge($cached->data, $data);
      }
      else {
        $registry = $this->initializeRegistry();
        $data = array_merge($registry, $data);
      }
      cache_set($this->cid, $data, $this->bin);
      if ($lock) {
        lock_release($lock_name);
      }
    }
  }
}

/**
 * Process a single implementation of hook_theme().
```

```
*
* @param $cache
*   The theme registry that will eventually be cached; It is an associative
*   array keyed by theme hooks, whose values are associative arrays describing
*   the hook:
*   - 'type': The passed-in $type.
*   - 'theme path': The passed-in $path.
*   - 'function': The name of the function generating output for this theme
*     hook. Either defined explicitly in hook_theme() or, if neither 'function'
*     nor 'template' is defined, then the default theme function name is used.
*     The default theme function name is the theme hook prefixed by either
*     'theme_' for modules or '$name_' for everything else. If 'function' is
*     defined, 'template' is not used.
*   - 'template': The filename of the template generating output for this
*     theme hook. The template is in the directory defined by the 'path' key of
*     hook_theme() or defaults to $path.
*   - 'variables': The variables for this theme hook as defined in
*     hook_theme(). If there is more than one implementation and 'variables' is
*     not specified in a later one, then the previous definition is kept.
*   - 'render element': The renderable element for this theme hook as defined
*     in hook_theme(). If there is more than one implementation and
*     'render element' is not specified in a later one, then the previous
*     definition is kept.
*   - 'preprocess functions': See theme() for detailed documentation.
*   - 'process functions': See theme() for detailed documentation.
* @param $name
*   The name of the module, theme engine, base theme engine, theme or base
*   theme implementing hook_theme().
* @param $type
```

```
 *   One of 'module', 'theme_engine', 'base_theme_engine', 'theme', or
 *   'base_theme'. Unlike regular hooks that can only be implemented by modules,
 *   each of these can implement hook_theme(). _theme_process_registry() is
 *   called in aforementioned order and new entries override older ones. For
 *   example, if a theme hook is both defined by a module and a theme, then the
 *   definition in the theme will be used.
 * @param $theme
 *   The loaded $theme object as returned from list_themes().
 * @param $path
 *   The directory where $name is. For example, modules/system or
 *   themes/bartik.
 *
 * @see theme()
 * @see _theme_process_registry()
 * @see hook_theme()
 * @see list_themes()
 */
function _theme_process_registry(&$cache, $name, $type, $theme, $path) {
  $result = array();

  // Processor functions work in two distinct phases with the process
  // functions always being executed after the preprocess functions.
  $variable_process_phases = array(
    'preprocess functions' => 'preprocess',
    'process functions'    => 'process',
  );

  $hook_defaults = array(
    'variables' => TRUE,
```

```
    'render element' => TRUE,
    'pattern' => TRUE,
    'base hook' => TRUE,
  );

  // Invoke the hook_theme() implementation, process what is returned, and
  // merge it into $cache.
  $function = $name . '_theme';
  if (function_exists($function)) {
    $result = $function($cache, $type, $theme, $path);
    foreach ($result as $hook => $info) {
      // When a theme or engine overrides a module's theme function
      // $result[$hook] will only contain key/value pairs for information being
      // overridden.  Pull the rest of the information from what was defined by
      // an earlier hook.

      // Fill in the type and path of the module, theme, or engine that
      // implements this theme function.
      $result[$hook]['type'] = $type;
      $result[$hook]['theme path'] = $path;

      // If function and file are omitted, default to standard naming
      // conventions.
      if (!isset($info['template']) && !isset($info['function'])) {
        $result[$hook]['function'] = ($type == 'module' ? 'theme_' : $name . '_') .
$hook;
      }

      if (isset($cache[$hook]['includes'])) {
```

```
  $result[$hook]['includes'] = $cache[$hook]['includes'];
}

// If the theme implementation defines a file, then also use the path
// that it defined. Otherwise use the default path. This allows
// system.module to declare theme functions on behalf of core .include
// files.
if (isset($info['file'])) {
  $include_file = isset($info['path']) ? $info['path'] : $path;
  $include_file .= '/' . $info['file'];
  include_once DRUPAL_ROOT . '/' . $include_file;
  $result[$hook]['includes'][] = $include_file;
}

// If the default keys are not set, use the default values registered
// by the module.
if (isset($cache[$hook])) {
  $result[$hook] += array_intersect_key($cache[$hook], $hook_defaults);
}

// The following apply only to theming hooks implemented as templates.
if (isset($info['template'])) {
  // Prepend the current theming path when none is set.
  if (!isset($info['path'])) {
    $result[$hook]['template'] = $path . '/' . $info['template'];
  }
}

// Allow variable processors for all theming hooks, whether the hook is
```

```
// implemented as a template or as a function.
foreach ($variable_process_phases as $phase_key => $phase) {
  // Check for existing variable processors. Ensure arrayness.
  if (!isset($info[$phase_key]) || !is_array($info[$phase_key])) {
    $info[$phase_key] = array();
    $prefixes = array();
    if ($type == 'module') {
      // Default variable processor prefix.
      $prefixes[] = 'template';
      // Add all modules so they can intervene with their own variable
      // processors. This allows them to provide variable processors even
      // if they are not the owner of the current hook.
      $prefixes += module_list();
    }
    elseif ($type == 'theme_engine' || $type == 'base_theme_engine') {
      // Theme engines get an extra set that come before the normally
      // named variable processors.
      $prefixes[] = $name . '_engine';
      // The theme engine registers on behalf of the theme using the
      // theme's name.
      $prefixes[] = $theme;
    }
    else {
      // This applies when the theme manually registers their own variable
      // processors.
      $prefixes[] = $name;
    }
    foreach ($prefixes as $prefix) {
      // Only use non-hook-specific variable processors for theming hooks
```

```
          // implemented as templates. See theme().
          if (isset($info['template']) && function_exists($prefix . '_' . $phase)) {
            $info[$phase_key][] = $prefix . '_' . $phase;
          }
          if (function_exists($prefix . '_' . $phase . '_' . $hook)) {
            $info[$phase_key][] = $prefix . '_' . $phase . '_' . $hook;
          }
        }
      }
      // Check for the override flag and prevent the cached variable
      // processors from being used. This allows themes or theme engines to
      // remove variable processors set earlier in the registry build.
      if (!empty($info['override ' . $phase_key])) {
        // Flag not needed inside the registry.
        unset($result[$hook]['override ' . $phase_key]);
      }
      elseif (isset($cache[$hook][$phase_key]) &&
is_array($cache[$hook][$phase_key])) {
        $info[$phase_key] = array_merge($cache[$hook][$phase_key],
$info[$phase_key]);
      }
      $result[$hook][$phase_key] = $info[$phase_key];
    }
  }

  // Merge the newly created theme hooks into the existing cache.
  $cache = $result + $cache;
}
```

```
  // Let themes have variable processors even if they didn't register a template.
  if ($type == 'theme' || $type == 'base_theme') {
    foreach ($cache as $hook => $info) {
      // Check only if not registered by the theme or engine.
      if (empty($result[$hook])) {
        foreach ($variable_process_phases as $phase_key => $phase) {
          if (!isset($info[$phase_key])) {
            $cache[$hook][$phase_key] = array();
          }
          // Only use non-hook-specific variable processors for theming hooks
          // implemented as templates. See theme().
          if (isset($info['template']) && function_exists($name . '_' . $phase)) {
            $cache[$hook][$phase_key][] = $name . '_' . $phase;
          }
          if (function_exists($name . '_' . $phase . '_' . $hook)) {
            $cache[$hook][$phase_key][] = $name . '_' . $phase . '_' . $hook;
            $cache[$hook]['theme path'] = $path;
          }
          // Ensure uniqueness.
          $cache[$hook][$phase_key] = array_unique($cache[$hook][$phase_key]);
        }
      }
    }
  }
}
```