Документация к проекту «анимация серии иллюстраций У.Блейка»

1. Об иллюстрациях и их авторе

Одними из лучших иллюстраций к «Божественной комедии» Данте Алигьери являются акварели и гравюры английского художника и поэта Уильяма Блейка (1757-1827), известного также [своими работами по библейской «Книге Иова](http://top-antropos.com/religion/hristianstvo/item/816-kniga-iova-william-blake)». Для того, чтобы выполнить иллюстрации к поэме Данте, Блейк в 67 лет начинает изучать итальянский язык. Художник начал работу над циклом иллюстраций к «Божественной комедии» за год до смерти и не успел закончить начатое. Сохранились иллюстрации (102), в том числе незаконченные, почти ко всем песням «Ада» (69 иллюстраций), частично были проиллюстрированы «Чистилище» (20 иллюстраций) и «Рай» (10 иллюстраций). 3 иллюстрации сохранились без указания песни.

Ознакомиться со всеми иллюстрациями можно, перейдя по ссылке:

<http://top-antropos.com/literature/item/838-william-blake-dante>

**Уильям Блейк** (англ. *William Blake*; 28 ноября 1757, Лондон — 12 августа 1827, Лондон) — английский поэт, художник и гравёр. Почти непризнанный при жизни, Блейк в настоящее время считается важной фигурой в истории поэзии и изобразительного искусства романтической эпохи.  
В 1826 году Блейк получает заказ на создание серии гравюр к Божественной комедии Данте. Но смерть Блейка в 1827 году не даёт ему реализовать свою смелую задумку до конца. Но даже немногие законченные иллюстрации были удостоены восхищения:

«Учитывая всю сложность содержания Божественной комедии, акварельные иллюстрации к ней, талантливо выполненные Блейком, находятся в ряду величайших достижений художника. Мастерство в области акварельной живописи достигло ещё более высокого уровня чем раньше, и использовано для невероятного эффекта при воссоздании абсолютно уникальной атмосферы каждого из трёх состояний бытия в поэме.» — David Bindman, «Blake as a Painter»

Иллюстрации Блейка к поэме не сопровождают описываемое буквально, они, скорее, заставляют критически пересмотреть происходящее, порой предоставляя новое видение духовных и моральных сторон произведения.  
Так как проекту не было суждено завершиться, замысел Блейка остался не совсем ясным. Некоторые же придерживаются того мнения, что вывод о нём можно сделать, только говоря в общем обо всей серии иллюстраций. А именно: они бросают вызов тексту, который сопровождают, оспаривая мнение автора: например, о сцене, где Гомер шествует с мечом и своими соратниками, Блейк пишет: «Всё в Божественной комедии говорит о том, что из-за своих тиранических идей Данте “сделал” Этот Мир из “Сотворения” и “Богини Природы”, но без участия Святого Духа». Возможно, Блейк не разделял и восхищения Данте поэзией древних греков, а также той несомненной радости, с которой он назначал и раздавал обвинения и наказания в Аду (о чём свидетельствует мрачный юмор некоторых песен поэмы).  
Однако Блейк разделял недоверие Данте к материализму и протест против развращённой природы власти. Он также получал большое удовольствие от возможности представить своё личное восприятие атмосферы поэмы визуально, путём иллюстрации. Даже появившееся у Блейка чувство приближения смерти не смогло отвлечь его от творчества, которым он был полностью поглощён. В это время он лихорадочно корпел над Адом Данте. Считается что он потратил один из последних шиллингов на карандаш, чтобы продолжить рисовать эскизы.

Узнать больше о биографии Уильяма Блейка и его творчестве можно на [сайте, целиком посвящённом ему](http://blake.sacrum.ru/).

Вся фоновая музыка является частью официального саундтрека к видеоигре и одноименному аниме «Dante's Inferno: An Animated Epic». Главные композиторы - Garry Schyman и Paul Gorman.

Для анимации мною были выбраны иллюстрации «Ада». На данный момент были созданы анимированные иллюстрации следующих пяти сцен:

* в сумеречном лесу;
* перед вратами ада;
* перед переправой через Ахерон;
* встреча с Гомером, Овидием, Горацием;
* встреча с Миносом.

Также для удобства переключения между этими иллюстрациями была написана страница – схема Ада. К сожалению, У. Блейк не создал иллюстрацию к «Божественной комедии», изображающую общее устройство ада по Данте. Поэтому была выбрана иллюстрация неизвестного автора, права на которую принадлежат [Unione Fiorentina of the Museo Casa di Dante](https://www.museocasadidante.it/).

В результате связь между страницами выглядит следующим образом:

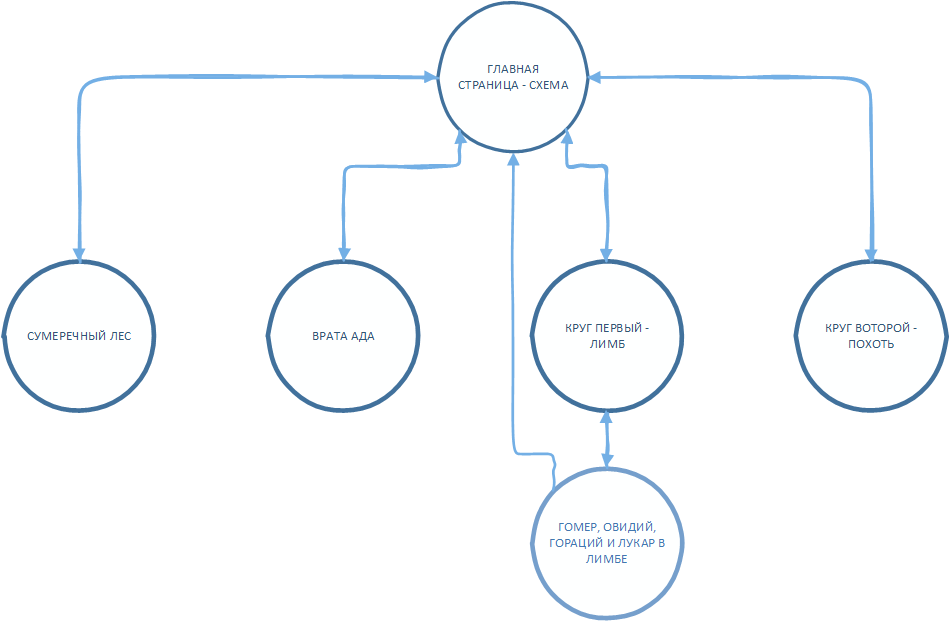


Схема связей между страницами

1. Страница – схема Ада

Путь к html файлу: dante\_inferno\_html\_css\_js\schema\_inferno\_demo.html

Данная иллюстрация доступна в интернете с подписями только на итальянском, поэтому на начальном этапе работы был выполнен перевод всего текста, присутствующего на изображении, на русский язык. Сделано это было для удобства восприятия информации.

На следующем этапе была поставлена задача реализовать интерактивную подсветку областей изображения, соответствующих тому или иному кругу ада. Также требовалась реализация возможности нажатия на подсвеченную область с дальнейшим переходом по ссылке, прикреплённой к области.

С обеими задачами помогли справиться [карты-изображения (image map)](http://htmlbook.ru/content/karty-izobrazheniya).

Карта изображений — это графический объект языка разметки HTML, связанный с изображением и содержащий специальные области (активные зоны), при нажатии на которые происходит переход по определённому URL (при помощи javascript можно установить другие действия). Использование карт изображений позволяет хранить несколько ссылок в одном изображении.  
Для добавления карты изображений на веб-страницу используется парный HTML-тег <map>, в атрибуте "name" которого указывается уникальный в пределах документа идентификатор карты. Внутрь него добавляются непарные теги <area>, каждый из которых описывает одну активную зону. Затем в тег <img>, который необходимо связать с картой, добавляется атрибут usemap, содержащий имя привязываемой карты. Значение атрибута должно начинаться со знака решётки #. При этом в теге <img> должны быть явно указаны размеры изображения.  
Внутри контейнера <map> располагается один или несколько элементов <area>, они задают форму области, её координаты, устанавливают адрес документа, на который следует сделать ссылку, а также всплывающую подсказку.  
Элемент <area> имеет следующие атрибуты.

* shape — определяет форму активной области. Форма может быть в виде окружности (circle), прямоугольника (rect), полигона (poly).
* alt — добавляет альтернативный текст для каждой области. Служит лишь комментарием для ссылки, поскольку на экран не выводится. Обязательный атрибут при наличии href.
* title — выводит всплывающую подсказку при наведении курсора на область.
* href — задаёт адрес документа, на который следует перейти, по своему действию аналогичен подобному атрибуту элемента <a>.
* coords — задаёт координаты активной области. Координаты отсчитываются в пикселях от левого верхнего угла изображения, которому соответствует значение 0, 0. Первое число является координатой по горизонтали, второе — по вертикали. Список координат зависит от формы области.

Для окружности задаются три числа — координаты центра круга и радиус:

1. < area shape = "circle" coords = "230, 340, 100" href = "circle.html" alt = "" >

Для прямоугольника — координаты левого верхнего и правого нижнего угла:

1. < area shape = "rect" coords = "24,18, 210, 56" href = "rect.html" alt = "" >

Для полигона задаются координаты его вершин.

В определении координат активных областей помогли следующие проекты:

* [Рисуем карту изображения мышкой](https://habrahabr.ru/post/142419/);
* [Пишем редактор карт изображений (html image map) на javascript / html5 / inline SVG](https://habrahabr.ru/post/152731/);
* [Онлайн генератор областей карты](http://en.crazysquirrel.ru/#!image_to_map_area).

Вот так выглядит часть кода, отвечающая за подсветку сумеречного леса и последующий переход к соответствующей интерактивной иллюстрации:

1. < img class = "map-all map" src = "dante\_inferno.png" usemap = "#schema\_inferno\_map" >
2. < map name = "schema\_inferno\_map" >
3. < area alt = "Duskwood" title = "Duskwood" shape = "poly"
4. coords = "336,180, 340,188, 361,188, 378,217, 393,225, 406,218, 432,242, 517,226, 588,200, 705,165, 720,170, 780,146, 768,120, 733,113, 713,122, 707,101, 700,111, 672,117, 663,102, 658,117, 650,121, 648,100, 638,120, 623,125, 617,136, 607,141, 605,132, 583,136, 587,148, 575,150, 576,142, 567,137, 550,143, 512,143, 501,140, 478,147, 455,156, 450,143, 440,156, 346,167"
5. href = "duskwood/duskwood.html" >
6. < /map>

После решения вопроса о подсветке и привязке ссылок захотелось получить возможность изменять вид подсветки. Сделать это получилось с помощью javascript библиотеки [jquery.maphilight.js](http://davidlynch.org/projects/maphilight/docs/). Библиотека позволяет задавать цвет и прозрачность самой подсветки, цвет и толщину обводки, различные параметры теней. Для работы этой библиотеки необходимо подключить библиотеку jQuery.   
Также хотелось сделать карту «резиновой», что стандартный html сделать, конечно, не позволяет. Пришлось обратиться к другой javascript библиотеке - [rwdImageMaps](https://github.com/stowball/jQuery-rwdImageMaps). Это проект, позволяющий масштабировать изображение в зависимости от размера окна браузера, основанный на image maps. Для работы требуется подключение jQuery.  
Но в оригинальном виде библиотеки maphilight и rwdImageMaps не работают вместе. С этой проблемой помог справиться [небольшой проект](http://i-rrv.ru/масштабируемая-карта-map-area-с-подсветкой/), в котором библиотека maphilight предложена в несколько изменённом варианте.

В результате был получен следующий код:

* Html дополнение, атрибуты тега area (фрагмент для области врад ада)

1. data - maphilight = "{&quot;strokeColor&quot;:&quot;C21B0F&quot;,&quot;strokeWidth&quot;:2 ,
2. &quot;fillColor&quot;:&quot;CFB53B&quot;,&quot;fillOpacity&quot;:0.4}"

* Javascript код, отвечающий за подключение и использование всех необходимых библиотек:

1. < script type = "text/javascript" src = "http://code.jquery.com/jquery1.9.1.min.js" >
2. < /script>
3. < script type = "text/javascript" src = "jquery.rwdImageMaps.min.js" > < /script>
4. < script type = "text/javascript" src = "jquery.maphilight.resize.min.js" > < /script>
5. < script >
6. jQuery(function() {
7. jQuery('img[usemap]').rwdImageMaps();
8. jQuery('.map').maphilight();
9. jQuery(window).resize(function() {
10. jQuery('.map').maphilight();
11. });
12. });
13. < /script>

Позже на странице была реализована плавная прокрутка вверх и вниз кнопками:

1. < div class = "go-up" title = "To Life" id = 'ToTop' > & #9650;</div>
2. <div class= "go-down" title = "To Hell" id = 'OnBottom' > & #9660;</div>
3. <script type= 'text/javascript' > $(function() {
4. if ($(window).scrollTop() >= "10") $("#ToTop").fadeIn("slow") $(window).scroll(function(){
5. if ($(window).scrollTop() <= "10") $("#ToTop").fadeOut("slow")
6. else $("#ToTop").fadeIn("slow")
7. });
8. if ($(window).scrollTop() <= "300") $("#OnBottom").fadeIn("slow") $(window).scroll(
9. function() {
10. if ($(window).scrollTop() >= "250") $("#OnBottom").fadeOut("slow")
11. else $("#OnBottom").fadeIn("slow")
12. });
13. $("#ToTop").click(function(){$("html,body").animate({scrollTop: 0}, "slow")})
14. $("#OnBottom").click(function(){$("html,body").animate({scrollTop: $(document).height()
15. }, "slow")})
16. });
17. < /script>

Также на страницу была добавлен фоновый звук:

1. < audio id = "myPlayer" autoplay loop >
2. < source src = "Redemption.ogg" type = "audio/ogg" >
3. < source src = "Redemption.mp3" type = "audio/mpeg" >
4. Ваш браузер не поддерживает теги audio. Необходимо обновить браузер!
5. < /audio>
6. Иллюстрация первая – «Данте и Вергилий в сумеречном лесу»

Путь к html файлу: dante\_inferno\_html\_css\_js\duskwood\duskwood.html

Данная иллюстрация соответствует Песни первой и изображает Вергилия, Данте и три зверя: рысь (символ сладострастия), льва (гордость) и волчицу (корыстолюбие).

Для усиления впечатления от иллюстрации было сделано следующее:

1. Звёзды начинают «мерцать» с момента загрузки страницы. Для создания такого эффекта звёзды были вырезаны, раскрашены в другой цвет и увеличены в размерах, затем был написан следующий код анимации:
2. @keyframes shining\_stars {
3. from {
4. background - image: url(stars.png);
5. }
6. 50 % {
7. background - image: url(shine\_stars.png);
8. }
9. to {
10. background - image: url(stars.png);
11. }
12. }
13. При клике мышкой на Данте, тот издаёт вскрик и вскидывает руки.

Руки были помещены на страницу с помощью использования пседвоэлементов before и after, родительским элементом является блок, отвечающий за тело Данте. Поверх этих трёх блоков была помещёна пустая ссылка со свойствами блока и с заданными размерами. При нажатии на неё за звук отвечает html тег <audio> с прописанной для него javascript функцией onclick в ссылке:

1. < a href = "#" onclick = "scream()" class = "element hidden\_dante"> < /a> <audio id="dante\_sound" preload="auto">
2. < source src = "Scream.mp3" type = "audio/mpeg">
3. < source src = "Scream.ogg" type = "audio/ogg">
4. < /audio>
5. …
6. <script>
7. var soundLinkDante = $("#dante\_sound")[0];
8. function scream() {
9. soundLinkDante.play();
10. };
11. < /script>

За анимацию рук отвечает css3-свойство transform:rotate(), условием применения которого является нажатие мыши на ссылку. Для отклика на нажатие использован псевдокласс focus:

1. .dante::before {  
       transform - origin: 86 % 82 % ;
2. transform: rotate(-40 deg);
3. transition - duration: 2 s;  
   }  
   .hidden\_dante:focus ~ .dante::before {
4. transform: rotate(0 deg);
5. }
6. По тому же принципу, что и для Данте, сделана анимация для Вергилия. При нажатии на пустую область тот «слетает» на место. Для отклика на нажатие использован псевдокласс focus. Движение задаётся css3-свойтвом transform:translate(xpx,ypx):
7. .hidden\_vergiliy:focus ~ .vergiliy {
8. transform: translate(-90 px, 523 px);
9. }

1. Звери на иллюстрации анимированы по одной схеме:  
   мерцание глаз и движение хвоста начинает происходить при наведении мыши на невидимый блок. Для отклика на наведение использован псевдокласс hover:
2. .hidden\_wolf:hover ~ .wolf\_eyes {
3. animation: shining\_wolf 3s infinite linear;
4. }
5. .hidden\_wolf:hover ~ .wolf\_tail {
6. animation: shaking\_wolf 10s infinite ease-in-out;
7. }

рычание также раздаётся при наведении на невидимый блок:

1. < div class = "element hidden\_wolf"> < /div>
2. < audio id = "wolf\_sound" preload = "auto">
3. < source src = "Wolfs\_growl.mp3" type = "audio/mpeg">
4. < source src = "Wolfs\_growl.ogg" type = "audio/ogg">
5. </audio>
7. <script>
8. var soundLinkWolf = $("#wolf\_sound")[0];
9. $(".hidden\_wolf").mouseenter(function() {
10. soundLinkWolf.play();
11. });
12. $(".hidden\_wolf").mouseleave(function() {
13. soundLinkWolf.pause();
14. });
15. </script>
16. Цветок и листва на дереве шевелятся «от ветра» при наведении мыши. Для создания плавной повторяющейся анимации было использовано css3-свойство animation-direction. Для отклика на наведение использован псевдокласс hover:
17. .tree:hover ~ .tree\_anim {
18. animation: wind\_tree 3 s infinite;
19. animation-direction: alternate;
20. }
21. .flover:hover {
22. animation: wind\_flover 5 s infinite;
23. animation-direction: alternate;
24. }

Звук ветра:

1. < div class = "element tree"> < /div>
2. < div class = "element flover"> < /div>
3. < audio id = "wind\_sound" preload = "auto" loop = "loop">
4. < source src = "Wind.ogg" type = "audio/ogg">
5. < source src = "Wind.mp3" type = "audio/mpeg">
6. </audio>
7. <script>
8. var soundLinkWind = $("#wind\_sound")[0];
9. $(".flover").mouseenter(function() {
10. soundLinkWind.play();
11. });
12. $(".flover").mouseleave(function() {
13. soundLinkWind.pause();
14. });
15. $(".tree").mouseenter(function() {
16. soundLinkWind.play();
17. });
18. $(".tree").mouseleave(function() {
19. soundLinkWind.pause();
20. });
21. </script>
22. С помощью линейного (вода) и радиального (небо) градиента и анимации изменения его прозрачности (css-свойство opacity) сделан эффект переливающегося рассветного неба. Блоки с градиентом являются пседвоэлементами before и after. Для плавности и непрерывности анимации было использовано css3-свойство animation-direction:
23. .sky\_water::before {
24. background: radial-gradient(ellipse at 462 px 388 px,
25. #fecd71, #d98594, #6396d8, # 7 a5c9c);
26. animation: sun\_sky 12000 ms infinite;
27. animation-direction: alternate;
28. }
29. .sky\_water::after {
30. background: linear-gradient(to bottom,
31. #7a5c9c 0%, #fec5b9 3%, #dc8998 15.15%, #718ec4 29.443%, # 787fa1 45.459%);
32. animation: sun\_water 12000 ms infinite;
33. animation-direction: alternate;
34. }
35. @keyframes sun\_sky {
36. from {
37. opacity: 0;
38. }
39. to {
40. opacity: 0.7;
41. }
42. }
43. @keyframes sun\_water {
44. from {
45. opacity: 0;
46. }
47. to {
48. opacity: 0.7;
49. }
50. }
51. Реализовано возвращение на страницу – схему ада. Использована пустая ссылка с блочными свойствами и фоновой картинкой. У элемента задана прозрачность и яркость с помощью css-свойства opacity и css3-свойтва filter:brightness(%) соответственно. Для обращения внимания на ссылку использована анимация из подключенной библиотеки [hover.css](https://ianlunn.github.io/Hover/), для ссылки добавлен специальный класс hvr-pulse:
52. <link rel = "stylesheet" type = "text/css"
53. href = "..//Hover-master/css/hover.css">
54. < a href = "..//schema\_inferno\_demo.html" class = "exit hvr-pulse"> </a>
55. .exit {
56. background: url(.. //Green\_back\_sketch.png) no-repeat;
57. background - size: contain;
58. opacity: 0.7;
59. filter: brightness(60%);
60. }
61. .exit:hover {
62. opacity: 1;
63. filter: brightness(90%);
64. }
65. Вставлен фоновый звук

Итого, количество эффектов, применённых к первой иллюстрации (без учёта фонового звука): 21.

1. Иллюстрация вторая – «Врата ада»

Путь к html файлу: dante\_inferno\_html\_css\_js\ hellgate\hellgate.html

Данная иллюстрация соответствует Песни третьей и изображает Вергилия и Данте у врат Ада.

Для усиления впечатления от иллюстрации было сделано следующее:

1. Листья деревьев по бокам от врат шелестят (звук) и трясутся при наведении мыши. Реализован звук при наведении. Была подключена библиотека [CSS Shake](https://elrumordelaluz.github.io/csshake/) и исльзована анимация shake-little из неё:
2. < link rel = "stylesheet" type = "text/css" href = "..//csshake.css">
3. .left\_tree:hover,
4. .right\_tree:hover {
5. animation: shake - little 5s infinite ease-in-out;
6. }
7. С момента загрузки страницы на заднем плане с заданным инервалом по реке проплывает Харон. Это сделано с помощью css3-свойства transform:tanslateX():
8. .boatman {
9. animation: working\_boatman 18s infinite linear;
10. }
11. @keyframes working\_boatman {
12. from {
13. transform: translateX(0);
14. }
15. to {
16. transform: translateX(500px);
17. }
18. }
19. С помощью линейного градиента была сделана мерцающая «завеса» на вратах. Для плавности анимации использовано css3-свойство animation-direction:
20. .gradiental {
21. background: linear-gradient(to top right,
22. #222020 0%, #3B2B2B 20%, #4C2020 40%, #511010 60%, #7A0000 80%);
23. animation-direction: alternate;
24. }
25. Сделана анимация слоёв за завесой: при наведении мыши они наползают друг на друга, слышны звуки страданий. Для этого был создан пустой блок, помещённый поверх всех остальных блоков. Был написан код для взаимодействия этого блока с каждым из трёх слоёв отдельно, прописаны задержки анимации для второго и третьего слоёв, чтобы создалось ощущение хаотичности движения:
26. .rush:hover ~ .top\_scene {
27. animation: top\_rush 2s infinite linear;
28. animation-direction: alternate;
29. }
30. .rush:hover ~ .middle\_scene {
31. animation: middle\_rush 2s infinite linear;
32. animation-delay: 500ms;
33. animation-direction: alternate;
34. }
35. .rush:hover ~ .bottom\_scene {
36. animation: bottom\_rush 1s infinite linear;
37. animation-delay: 1s;
38. animation-direction: alternate;
39. }
40. При нажатии на табличку над вратами, на ней постепенно проявляется надпись «Оставь надежду, всяк сюда входящий», мерцающая после проявления, и раздаётся зловещий шёпот.   
    Для усиления эффекта от текста использовано css-свойство text-shadow, выбранный текст импортирован из [Google Fonts](https://fonts.google.com).
41. <a href = "#" class = "element hell\_text" onclick = "hell\_text()" > Оставь надежду, < br >всяк сюда входящий </a>
42. < audio id = "text\_sound" preload = "auto">
43. <source src = "for\_text.mp3" type = "audio/mpeg">
44. <source src = "for\_text.ogg" type = "audio/ogg" >
45. </audio>
46. <script type = "text/javascript">
47. var soundText = $("#text\_sound")[0];
48. function hell\_text() {
49. soundText.play();
50. };
51. </script>

1. @import url('https://fonts.googleapis.com/css?family=Marck+Script&subset=cyrillic');
2. .hell\_text {
3. font-family: 'Marck Script', cursive;
4. text-shadow: -1px -1px 5px #E71C04;
5. }
6. При нажатии на пустой «порог» перед вратами на иллюстрацию вплывает Вергилий, а следом за ним, с небольшой задержкой, и Данте. Сделать это с помощью javascript, т.к. на css невозможно реализовать однократную анимацию/переход так, чтобы элемент потом не вернулся в исходное состояние. В данной ситуации уместным было появление без последующего исчезновения.
7. <a href="#" class="element emergence"
8. onclick="hands\_up\_and\_come\_here()"> </a>
9. <div class="element hg\_dante" id="hg\_d"> </div>
10. <div class="element hg\_dante\_hand" id="hg\_d\_h"> </div>
11. <div class="element hg\_vergiliy"  id="hg\_v"> </div>
12. <div class="element hg\_vergiliy\_hand" id="hg\_v\_h"> </div>
13. <script type="text/javascript">
14. function hands\_up\_and\_come\_here() {
15. document.getElementById("hg\_d").style.transform = "translate(168px, -425px)";
16. document.getElementById("hg\_d\_h").style.transform = "translate(168px,
17. -425px)";
18. document.getElementById("hg\_v").style.transform = "translate(222px, -415px)";
19. document.getElementById("hg\_v\_h").style.transform = "translate(222px,
20. -415px)";
21. }
22. </script>
23. Реализовано возвращение на страницу – схему ада аналогично реализации на странице «duskwood».
24. Вставлен фоновый звук

Итого, количество эффектов, применённых ко второй иллюстрации (без учёта фонового звука): 15.

1. Иллюстрация третья – «Лимб, ожидание переправы через реку Ахерон»

Путь к html файлу: dante\_inferno\_html\_css\_js \round\_one\_Limb\ Limb.html

Данная иллюстрация соответствует Песни третьей и изображает Харона и души, собирающиеся пересечь реку Ахерон.

Для усиления впечатления от иллюстрации было сделано следующее:

1. Души грешников вверху сцены при наведении нах мыши начинают раскачиваться из стороны в сторону и горестно стенают.
2. При нажатии мыши на человека на скале, тот начинает лезть вверх, срывается, снова лезет, и так бесконечно:
3. .climber {
4. transform-origin: 100% 100%;
5. }
6. .hidden\_climber:focus ~ .climber {
7. animation: climb\_man\_climb 8 s infinite linear;
8. }
9. @keyframes climb\_man\_climb {
10. /\*up\*/
11. from {
12. transform: translate(0px, 0px) rotate(0deg);
13. }
14. 10% {
15. transform: translate(-7px, -5px) rotate(6deg);
16. }
17. 20% {
18. transform: translate(0px, -20px) rotate(0deg);
19. }
20. 30% {
21. transform: translate(7px, -40px) rotate(-4deg);
22. }
23. 40% {
24. transform: translate(-5px, -50px) rotate(0deg);
25. }
26. 50% {
27. transform: translate(0px, -60px) rotate(4deg);
28. }
29. 60% {
30. transform: translate(6px, -65px) rotate(-2deg);
31. }
32. 70% {
33. transform: translate(0px, -70px) rotate(0deg);
34. }
35. 80% {
36. transform: translate(-7px, -73px) rotate(3deg);
37. }
38. /\*fall\*/
39. 85% {
40. transform: translate(6px, -2px) rotate(4.5deg);
41. }
42. 90% {
43. transform: translate(6px, -2px) rotate(4.5deg);
44. }
45. 93% {
46. transform: translate(3px, -1px) rotate(2.25deg);
47. }
48. 96% {
49. transform: translate(0px, 0px) rotate(0deg);
50. }
51. to {
52. transform: translate(0px, 0px) rotate(0deg);
53. }
54. }
55. При наведении мыши на знаменосца тот начинает раскачиваться, махать штандартом. Для более естественного эффекта использованы css-свойство transform-origin и временная функция animation-timing-function, заданная кривой Безье, в краткой записи animation:
56. .gonfalonier {
57. transform-origin: 68.4% 59.89%;
58. }
59. .gonfalonier:hover {
60. animation: attention 4.5s infinite cubic-bezier(0.47, 0, 0.745, 0.715);
61. animation-direction: alternate;
62. }
63. При загрузке страницы Харон начинает перемещаться от берега к берегу. Если навести на него мышь, он остановится. Это сделано с помощью css-свойства animation-play-state:
64. .charon:hover {
65. animation-play-state: paused;
66. }
67. Сделан анимированный градиентный фон, аналогично анимации на странице «duskwood».
68. При наведении на рой пчёл те двигаются и жужжат.
69. Реализован переход на связанную по смыслу страницу - Гомер, Гораций, Овидий и Лукан в Лимбе (первый круг Ада). В этих целях использована ссылка с блочными свойствами, в обычном состоянии прозрачная. При наведении на неё, её прозрачность меняется, она начинает дрожать. Для этого к ссылке добавлен специальный класс hvr-buzz библиотеки hover css.
70. <a class="element meadow hvr-buzz" href="hgol.html"> </a>
71. .meadow {
72. opacity: 0;
73. }
74. .meadow:hover {
75. opacity: 0.3;
76. }
77. Реализовано возвращение на страницу – схему ада аналогично реализации на странице «duskwood».
78. Раз в какое-то время слышен шум реки. Это сделано с помощью html тега <audio> и javascript функций setInterval и setTimeout:
79. <script type="text/javascript"> setInterval(function() {
80. var soundRiver = $("#river\_sound")[0];
81. soundRiver.play();
82. setTimeout(function() {
83. soundRiver.pause();
84. }, 10000);
85. }, 15000);
86. </script>
87. Вставлен фоновый звук.

Итого, количество эффектов, применённых к третьей иллюстрации (без учёта фонового звука): 13.

1. Иллюстрация четвёртая – «Гомер, Гораций, Овидий и Лукан в Лимбе»

Путь к html файлу: dante\_inferno\_html\_css\_js \round\_one\_Limb\ hgol.html

Данная иллюстрация соответствует Песни четвёртой и иллюстрирует пребывание Гомера, Горация, Овидия и Лукана в Лимбе.

Для усиления впечатления от иллюстрации было сделано следующее:

1. При наведении на кроны деревьев листья на них трясутся и шелестят (звук). Анимация реализована по тому же принципу, что и для листьев в «hellgate». Использована библиотека CSS Shake.
2. При нажатии на огонь на возвышении он вспыхивает и потрескивает. В качестве контейнера использована пустая ссылка с блочными свойствами и фоном. Сделано с помощью замены фонового изображения ссылки.
3. При нажатии на небо медленно появляются души грешников. Сделано с помощью css-свойства opacity.
4. При наведении на центральное облако раздаются раскаты грома, облако мигает, как от молний. Последнее сделано с помощью быстрой, молниеносной смены значений свойства opacity во время анимации:
5. @keyframes flash {
6. from {
7. opacity: 0;
8. }
9. 10% {
10. opacity: 0.5;
11. }
12. 11% {
13. opacity: 0.8;
14. }
15. 12% {
16. opacity: 0.5;
17. }
18. 20% {
19. opacity: 0.3;
20. }
21. 21% {
22. opacity: 0.6;
23. }
24. 23% {
25. opacity: 0.3;
26. }
27. 30% {
28. opacity: 0.6;
29. }
30. 31% {
31. opacity: 0.9;
32. }
33. 32% {
34. opacity: 0.6;
35. }
36. 40% {
37. opacity: 0.2;
38. }
39. 41% {
40. opacity: 0.5;
41. }
42. 42% {
43. opacity: 0.2;
44. }
45. 50% {
46. opacity: 0.2;
47. }
48. 51% {
49. opacity: 0.8;
50. }
51. 53% {
52. opacity: 0.2;
53. }
54. }
55. При наведении на человека на краю скалы он падает вниз, раздаётся звук падения, человек возвращается на своё место на скале, и так бесконечно. Задержка звука сделана с помощью javascript функции setTimeout и clearTimeout:
56. <script type="text/javascript">
57. var soundFall=$("#sound\_fall")[0];
58. var timerId;
59. $(".hidden\_suicide").mouseenter(function() {
60. timerId = setTimeout(function() {
61. soundFall.play();
62. }, 3000);
63. });
64. $(".hidden\_suicide").mouseleave(function() {
65. clearTimeout(timerId);
66. soundFall.pause();
67. });
68. </script>
69. При наведении на пустующую поляну раздаются голоса и по очереди с помощью переходов css transition появляются Гомер, Гораций, Овидий и Лукан, сделаны задержки для них переходов:
70. .scientist.first {
71. transition-delay: 1s;
72. }.scientist.second {
73. transition-delay: 2s;
74. }.scientist.third {
75. transition-delay: 3s;
76. }.scientist.fourth {
77. transition-delay: 4s;
78. }
79. Сделано возвращение на страницу «Limb». Использована библиотека hover css, к ссылке подключён специальный класс hvr-wobble-bottom:
80. <a class="element hvr-wobble-bottom to\_limb" href="Limb.html"> </a>

1. Реализовано возвращение на страницу – схему ада аналогично реализации на странице «duskwood».
2. Вставлен фоновый звук

Итого, количество эффектов, применённых к четвёртой иллюстрации (без учёта фонового звука): 20.

1. Иллюстрация пятая – «Круг второй, страж Минос»
2. Общие эффекты и свойства
3. С целью улучшения кроссбраузерности для всех страниц был использован [автопрефиксёр](https://autoprefixer.github.io/ru/). Автопрефиксёр использует данные о популярности браузеров и поддержке ими вендорных префиксов. Опираясь на эту информацию, он расставляет и убирает префиксы.
4. Для всех страниц, кроме главной – схемы, используется css3-свойство box-sizing со значением border-box, которое позволяет не учитывать в размерах блока отступы и границы. Таким образом, указывая размеры блоку, вы указываете их всему блоку, а не содержимому. Это делает процесс расчёта размеров более лёгким.
5. \* {
6. box-sizing: border-box;
7. }
8. Ко всем страницам сделаны лоадеры. Они используются для того, чтобы длительная загрузка страницы (из-за большого количества качественной графики и некоторого количества скриптов и подключаемых файлов) происходила приятно визуально.

* Сам лоадер выполнен на чёрном непрозрачном фоне, перекрывающем загружающееся основное содержимое.
* Анимированные элементы лоадера сделаны с помощью маркированного списка <ul>, у которого убраны маркеры.
* Надписи сделаны с помощью псевдоэлементов before и after блочно-строчных элементов <span>.
* Шрифт для текста импортирован из Google Fonts.
* Для размещения обозначений сумеречного леса, врат ада и первого круга использованы css-свойства float: left/right и clear: both.
* Обращение к элементам списка происходит с помощью псевдоклассов first-child, last-child и htn-child().
* Анимация сделана с помощью css3-трансформации transform: translateY().

1. <div class="loader\_area">
2. <div class = "loader">
3. <span class = "round"> </span>
4. <span class = "standart"> </span>
5. <ul class = "scheme">
6. < li > < /li>
7. < li > < /li>
8. < li > < /li>
9. < li > < /li>
10. < li > < /li>
11. < li > < /li>
12. < li > < /li>
13. < li > < /li>
14. < li > < /li>
15. < li > < /li>
16. < li > < /li>
17. </ul>
18. </div>
19. </div>
20. @import url('https://fonts.googleapis.com/css?family=Gloria+Hallelujah|Space+Mono');
21. .loader\_area {
22. position: fixed;
23. overflow: hidden;
24. left: 0;
25. top: 0;
26. right: 0;
27. bottom: 0;
28. z-index: 100000;
29. width: 100%;
30. height: 100%;
31. background-color: #000000;
32. }
33. .round {
34. letter-spacing: 0.08em;
35. font-family: 'Gloria Hallelujah', cursive;
36. }
37. .round:before {
38. content: "ROUND ONE - LIMB";
39. }
40. .standart {
41. font-family: 'Space Mono', monospace;
42. letter-spacing:0.2em;
43. }
44. .standart:before {
45. content:"LOADING...";
46. }
47. .scheme li:first-child {
48. float:left;
49. }
50. .scheme li:nth-child(2) {
51. float: right;
52. }
53. .scheme li:nth-child(3) {
54. clear:both;
55. }
57. @keyframes first {
58. from {-webkit-transform: translateY(297px);transform: translateY(297px);}
59. 18.18% {-webkit-transform: translateY(297px);transform: translateY(297px);}
60. 27.27% {-webkit-transform: translateY(0px);transform: translateY(0px);}
61. }

Для того, чтобы лоадер исчезал после полной загрузки основного содержимого страницы (но не раньше, чем анимация лоадера покажется полностью один раз) написан следующий скрипт:

1. < script >
2. var waiting = new Promise(function(resolve) {
3. setTimeout(resolve, 6251);
4. });
5. $(window).on('load', function() {
6. waiting.then(function() {
7. var $preloader = $('.loader\_area'),
8. $loader = $preloader.find('.loader');
9. $loader.fadeOut();
10. $preloader.delay(350).fadeOut('slow');
11. });
12. });
13. < /script>