Руководство: Фреймы в TYPO3

Revised February 2001

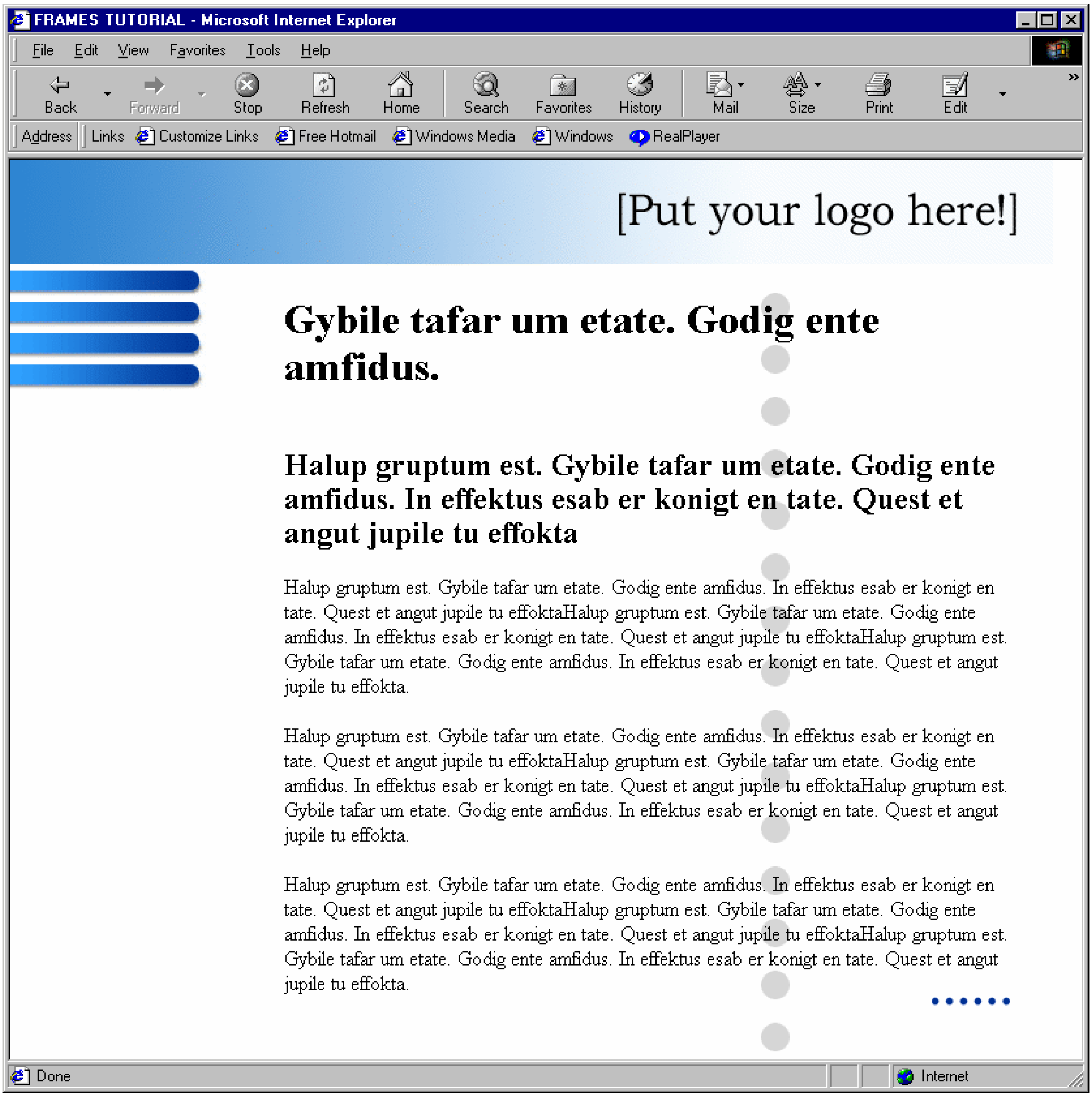
Оглавление

Настройка frameset и графического меню с фоновым изображением.

В этом руководстве предполагается, что вы знакомы с двумя документами: GoLive and References. Эти руководства дают основное понимание всего, что описано далее.

В этом руководстве мы разберем, как создавать дизайн, основанный на фреймах. Мы также создадим графическое меню и в конце подготовим без фреймовую версию этого сайта для старых броузеров.

Далее вы увидите пример того, к чему мы будем стремиться:



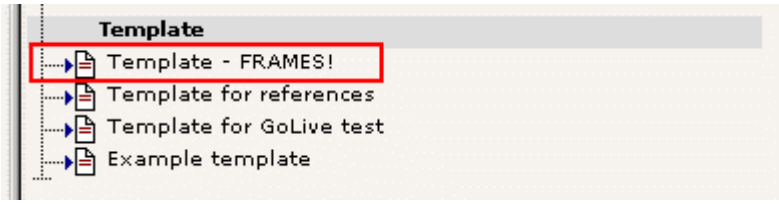
Прежде всего разархивируйте файлы в директорию "fileadmin/frames/" этого сайта.

Главное, вы должны подготовить шаблон, как вы делали это в руководстве “GoLive”. Действительно, вы можете скопировать точную копию этой записи.

В любом случае, очистите поле “Constants” и положите это в поле “Setup” - копию шаблонной записи "GoLive":

```
page = PAGE
page.typeNum = 0

page.10 = TEXT
page.10.value = Hello World
```

Так же поменяйте заголовки и убедитесь, что шаблон является первым шаблоном в списке!



Вы удвидите простой текст "Hello World" во frontend, когда посмотрите. Если нет, пожалуйста, вспомните об "Clear All Cache" в меню... :-)

Фреймы

Просмотрите беглым взглядом index.html:

```
<FRAMESET rows="85,*" framespacing="0" bordercolor="#FFFFFF" frameborder="0" border="0">
      <FRAME name="Top" src="top.html" frameborder="0" noresize marginheight="0" border="0"
scrolling="NO">
      <FRAMESET cols="170,*" framespacing="0" bordercolor="#FFFFFF" frameborder="0" border="0"
marginheight="0" border="0">
            <FRAME name="Menu" src="menu.html" frameborder="0" noresize marginheight="0" border="0"
scrolling="NO">
            <FRAME name="Main" src="content.html" frameborder="0" marginheight="0" border="0"
scrolling="AUTO">
      </FRAMESET>
</FRAMESET>
```

Это тот самый фрейм сет, который мы будет создавать.

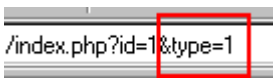
Первое, рассмотрим некоторые факты здесь: Typo3 использует "id" для переадресации на простую страницу. Но когда мы попадаем на определенную страницу, мы в действительности заставляем Typo3 генерировать 6 страниц (5); два framesets, top-page, menu-page и content-page – по одной в каждом фрейме. Теперь как это делается?

Дело в том, что Typo3 использует не только переменную “id”, но также и переменную “type”. Type-значение определяет, какой *TypoScript PAGE-объект* будет управлять от рисовкой страницы! Если “type” не задать, то по дефолту он равен нулю (0). И **каждое** определение страницы (page-definition) в Typo должно иметь собственное значение “type”.

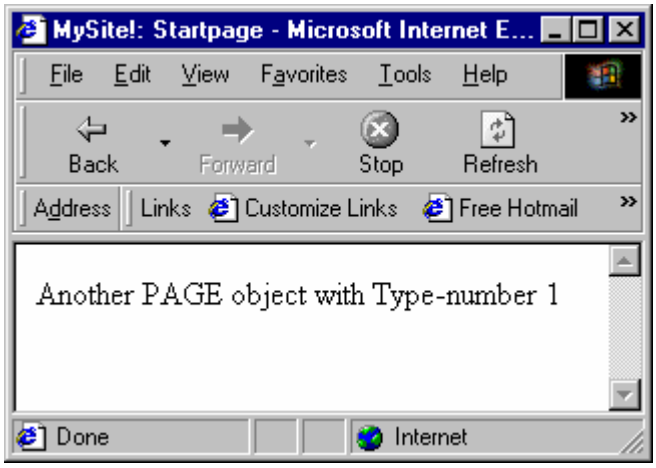
OK, попробуем это (положите код в поле “Setup”):

```
page = PAGE
page.typeNum = 0
page.10 = TEXT
page.10.value = Hello World

another_page_object = PAGE
another_page_object.typeNum = 1
another_page_object.10 = TEXT
another_page_object.10.value = Another PAGE object with Type-number 1
```

... и модифицируйте url до:



... и после попадания на frontend вы увидите:



Попробуйте установить "&type=1" обратно к нулю. Так же попробуйте установить его равным "2" или "3".

Как вы можете видеть, каждое type-значение определяет, какой PAGE-объект в TypoScript шаблоне отвечает за прорисовку страницы.

Что мы должны сделать, чтобы определить три (3) страницы, "content", "menu" и "top". Мы должны зарезервировать type-значение 0 (нуль) для frameset и в соответствии со стандартами шаблонов TypoScript (следование стандартам сделает вашу жизнь разработчика проще!), мы будем использовать "page", как имя фрейма, который содержит основной контент (main-content).

Очистите “Setup”-поле и положите в него:

```
# Definition of the page-objects
page = PAGE
menu = PAGE
top = PAGE

page.typeNum = 1
menu.typeNum = 2
top.typeNum = 3


# Defining the content-frame
page.10 = TEXT
page.10.value = The page-frame

# Defining the menu-frame
menu.10 = TEXT
menu.10.value = The menu-frame

# Defining the top-frame
top.10 = TEXT
top.10.value = The top-frame
```

Когда вы окажетесь во frontend, у вас будет возможность проверить создание трех новых PAGE-объектов, которы указаны в шаблоне. Опять попробуйте type-значения 1, 2 и 3!

Пожалуйста, бросьте беглый взгляд на PAGE-объект в документе TSRef. Посмотрите на доступные свойства этого объекта.

Теперь мы готовы положить три объекта в каждый фрейм.

Framesets

Frameset является действительно страницей. Так, добавьте это:

```
outer_frameset = PAGE
outer_frameset.typeNum = 0
```

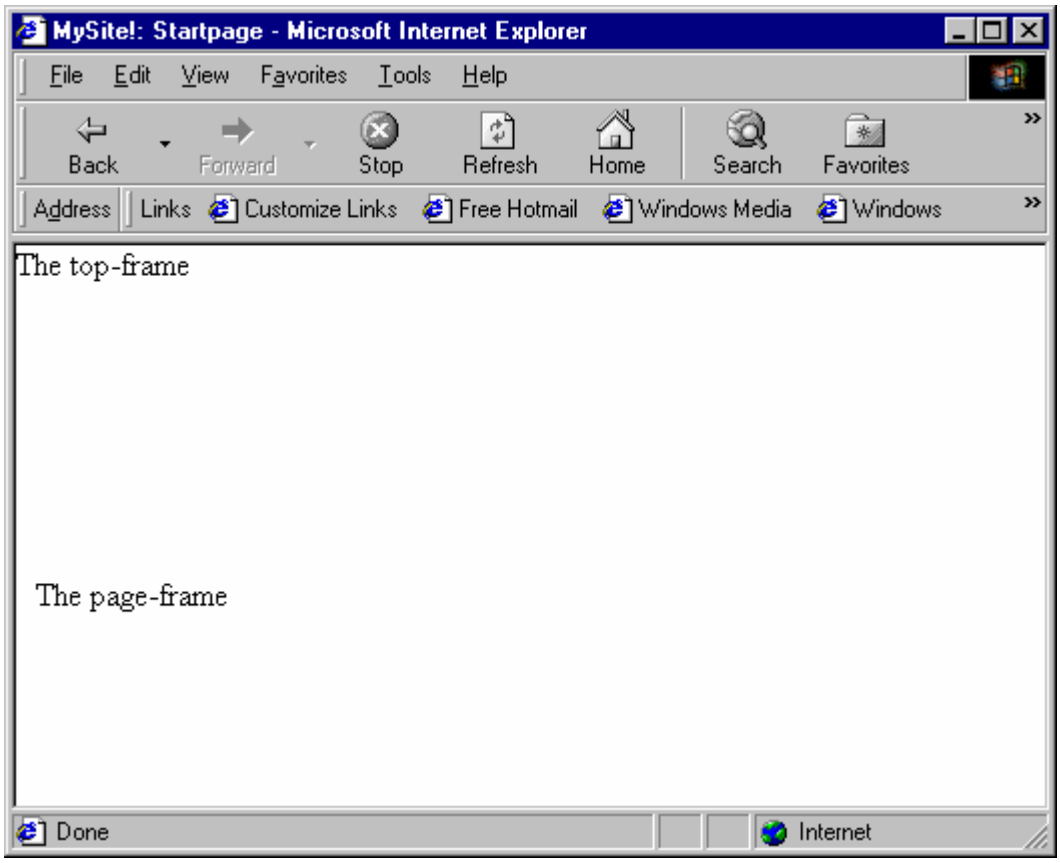
Посмотрев в документ TSRef, мы увидим, что PAGE-объект имеет свойство, "frameSet", которое делает страницей frameset и не простой страницей. Мы также уясним, что свойство "frameSet" в действительности FRAMESET-объект (обычно записывается "->FRAMESET"). Обратимся к описанию этого объекта в TSRef.

bgImg	imgResource	Background image on the page. This is automatically added to the body-tag.	
frameSet	->FRAMESET	If any properties is set to this property, the page is made into a frameset.	
meta	->META		
noLinkUnderline	boolean	Disables link-underlining. Uses in-document stylesheet.	

Нам улыбнулась удача – здесь маленький пример этого! Мы можем использовать пример (после маленькой адаптации):

```
# Defining the outer frameset
outer_frameset.frameSet.rows = 150,*
outer_frameset.frameSet.params = border="0" framespacing="0" frameborder="NO"
outer_frameset.frameSet {
  1 = FRAME
  1.obj = top
  1.params = scrolling="NO" noresize frameborder="NO" marginwidth="0" marginheight="0"
  2 = FRAME
  2.obj = page
  2.params = scrolling="AUTO" noresize frameborder="NO"
}
```

... и вот что мы получим:



Так, посмотрим FRAMESET объект в TSRef и нам откроется, что тип данных массива "1,2,3,..." в ".frameSet"-свойстве - "frameObj". Тип данных... хмм... несколько страниц назад, и там лист помощи: "frameObj" значит, что вы можете определить эти цифровые значения свойства объекта FRAMESET, как ... “FRAMESET” или “FRAME”.

Data types: Object types

This is some "data-types" that might be mentioned and valid values are shown here below:

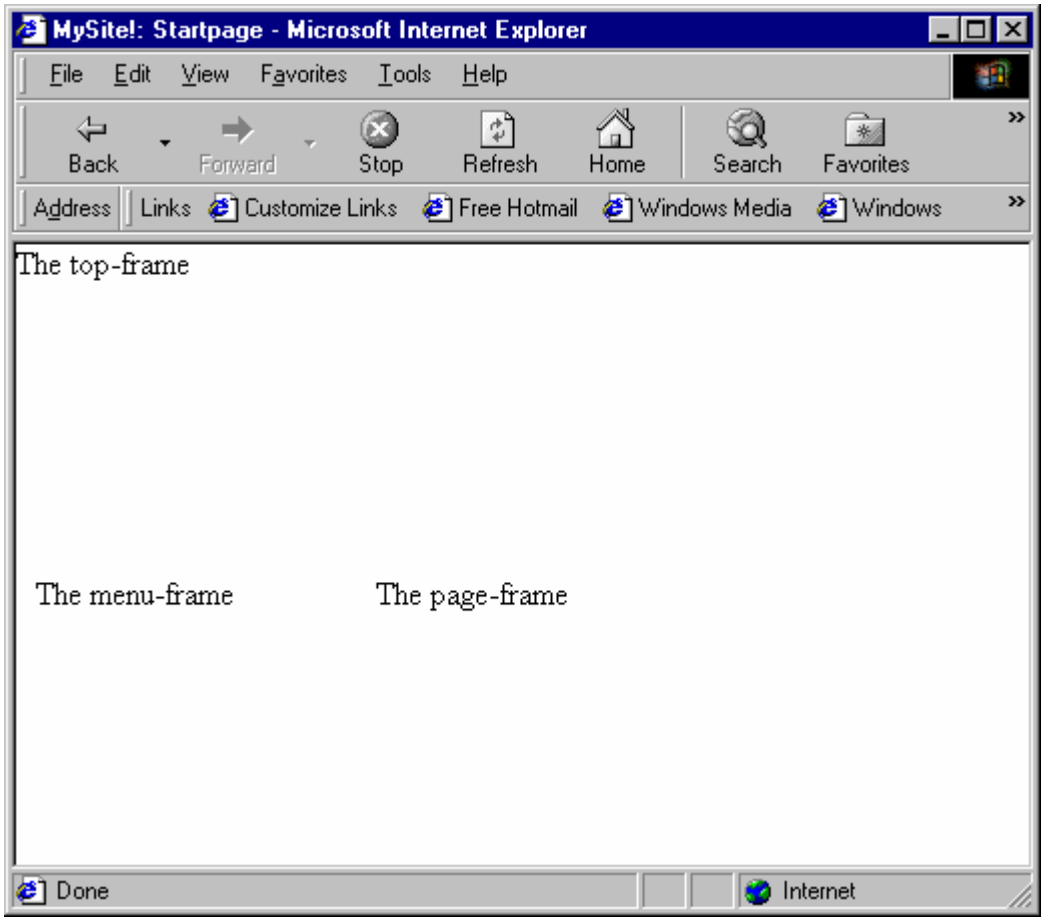
Datatype:	Examples:	Comment:	Default:
contentObj		HTML / TEXT / IMAGE .... (see "ContentObject" alsp mentioned "cObjects")	
frameObj		FRAMESET / FRAME	
menuObj		GMENU / TMENU / IMGMENU / JSMENU	
GifBuilderObj		TEXT / SHADOW / OUTLINE / EMBOSS / BOX / IMAGE / EFFECT	

Посмотрим на пример из TSRef – пример, который мы испытывали – мы увидим, что два объекта, 1 и 2 определены как FRAME-объекты. Очевидно, они в будут определены для содержания "top" и "page" PAGE-объектов. (Смотрите выше, а также в TSRef од FRAME).

Соответственно “frameset” мы оправляемся выполнять, предполагается быть уплотненным frameset, поэтому мы нуждаемся,чтобы один их этих объектов был “frameset” и не “frame”. Это будет сделано так:

```
# Defining the outer frameset
outer_frameset.frameSet.rows = 150,*
outer_frameset.frameSet.params = border="0" framespacing="0" frameborder="NO"
outer_frameset.frameSet {
  1 = FRAME
  1.obj = top
  1.params = scrolling="NO" noresize frameborder="NO" marginwidth="0" marginheight="0"
  2 = FRAMESET
  2.cols = 170, *
  2 {
    1 = FRAME
    1.obj = menu
    2 = FRAME
    2.obj = page
  }
}
```

... и мы получим следующее:



Вот так. Одну вещь нам надо добавить к параметрам оригинального HTML-кода frameset:

HTML:

```
<FRAMESET rows="85,*" framespacing="0" bordercolor="#FFFFFF" frameborder="0" border="0">
      <FRAME name="Top" src="top.html" frameborder="0" noresize marginheight="0" border="0"
scrolling="NO">
      <FRAMESET cols="170,*" framespacing="0" bordercolor="#FFFFFF" frameborder="0" border="0"
marginheight="0" border="0">
            <FRAME name="Menu" src="menu.html" frameborder="0" noresize marginheight="0" border="0"
scrolling="NO">
            <FRAME name="Main" src="content.html" frameborder="0" marginheight="0" border="0"
scrolling="AUTO">
      </FRAMESET>
</FRAMESET>
```

TypoScript:

```
outer_frameset.frameSet.rows = 85,*
outer_frameset.frameSet.params = framespacing="0" bordercolor="#FFFFFF" frameborder="0" border="0"
outer_frameset.frameSet {
  1 = FRAME
  1.obj = top
  1.params = frameborder="0" noresize marginheight="0" border="0" scrolling="NO"
  2 = FRAMESET
  2.params = framespacing="0" bordercolor="#FFFFFF" frameborder="0" border="0" marginheight="0"
border="0"
  2.cols = 170, *
  2 {
    1 = FRAME
    1.obj = menu
    1.params = frameborder="0" noresize marginheight="0" border="0" scrolling="NO"
    2 = FRAME
    2.obj = page
    2.params = frameborder="0" marginheight="0" border="0" scrolling="AUTO"
  }
}
```

Посмотрим в HTML исходник кода frameset, полученный генерацией Typo3, мы увидим это:

```
<FRAMESET rows="85,*" framespacing="0" bordercolor="#FFFFFF" frameborder="0" border="0">
<FRAME src="index.php?id=1&type=3" name="top" frameborder="0" noresize marginheight="0" border="0"
```

```
scrolling="NO">
<FRAMESET cols="170, *" framespacing="0" bordercolor="#FFFFFF" frameborder="0" border="0"
marginheight="0" border="0">
<FRAME src="index.php?id=1&type=2" name="menu" frameborder="0" noresize marginheight="0" border="0"
scrolling="NO">
<FRAME src="index.php?id=1&type=1" name="page" frameborder="0" marginheight="0" border="0"
scrolling="AUTO">
</FRAMESET>
</FRAMESET>
<noframes>
<body bgcolor="#FFFFFF">


</body>
</noframes></html>
```

Заметьте, как Typo3 автоматически вставляет правильный ссылки на "различные страницы" с одинаковым “id”.

Почему именно так?

Вы можете спросить, почему это сделано именно таким путем. Почему не позволено статическому html-документу быть frameset для включения в Typo3-страницу?

Ок, для этого есть один важный резон. Создания framesets таким способом позволяет вам создавать ссылки на различные страницы вашего Typo3 сайта, которые будут точно определять, что frameset прекрасно сгенерирован! Скажем другими словами: Если вы обратитесь к этой странице: "index.php?id=1" или "...id=2" или "id=2384" вы всегда получите правильную страницу с любым “frameset” выведенным около... (против получения уже контент-страницы в content-frame!). Если вы позволите "index.html" быть вашим frameset, вы никогда не свяжете внешнюю ссылку с определенной страницей сайта, потому что "index.html" будет всегда начинаться с главной страницы и меню.

Ок, если вы не поняли, не расстраивайтесь. Это большое дело, просто помните о подобном...!

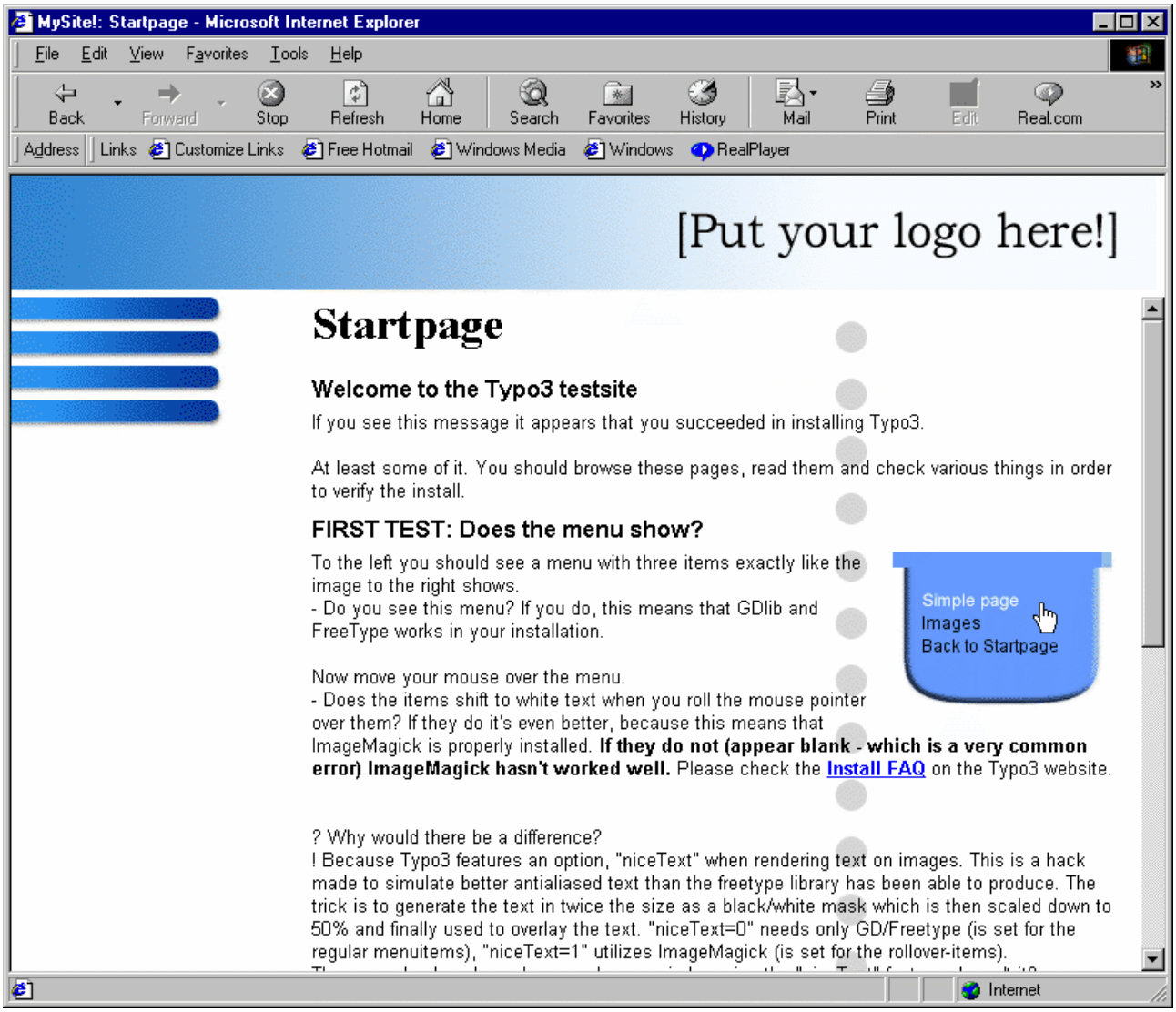
"Top"-, "content"- и "menu" PAGE-объекты

Поскольку сущность этого обсуждалась в предыдущих руководствах, я опущу описание их здесь и сразу представлю код работающий код TypoScript:

```
# ************************
# Defining the content-frame, "page"
# ************************

# Header code, stylesheet
page.headerData.10 = TEMPLATE
page.headerData.10 {
  template = FILE
  template.file = fileadmin/frames/content.html
  workOnSubpart = HEADER_CODE
}

# Bodytag
page.bodyTag = <BODY bgcolor="#ffffff" background="fileadmin/frames/background.gif"/>

# Page-content
page.10 = TEMPLATE
page.10 {
  template = FILE
  template.file = fileadmin/frames/content.html
  workOnSubpart = DOCUMENT_BODY
  subparts.CONTENT < styles.content.get
  subparts.PAGE_HEADER = TEXT
  subparts.PAGE_HEADER.field = title
}


# ************************
# Defining the top-frame, "top"
# ************************

# Header code, stylesheet
top.headerData.10 = TEMPLATE
top.headerData.10 {
  template = FILE
  template.file = fileadmin/frames/top.html
  workOnSubpart = HEADER_CODE
}

# Bodytag
top.bodyTag = <BODY bgcolor="#ffffff" leftmargin="0" topmargin="0">

# Page-content
```

```
top.10 = TEMPLATE
top.10 {
  template = FILE
  template.file = fileadmin/frames/top.html
  workOnSubpart = DOCUMENT_BODY
}

# ************************
# Defining the menu-frame, "menu"
# ************************

# Header code, stylesheet
menu.headerData.10 = TEMPLATE
menu.headerData.10 {
  template = FILE
  template.file = fileadmin/frames/menu.html
  workOnSubpart = HEADER_CODE
}

# Bodytag
menu.bodyTag = <BODY bgcolor="#ffffff">

# Page-content
menu.10 = TEMPLATE
menu.10 {
  template = FILE
  template.file = fileadmin/frames/menu.html
  workOnSubpart = DOCUMENT_BODY
}
```

Это будет выглядеть подобно:



Графическое меню

Эта серия руководств имеет целью сделать как можно больше без TypoScript, используя TypoScript только для установки наиболее общих основных частей сайта. Вам нет необходимости иметь включенный HTML-документ для создания сайта. Все может быть сделано внутри.

Отдельно от общего отображения страничного контента стоит другой вид элементов, который нуждается в TypoScript. Это динамические меню!

Последняя сложная задача – это графическое меню. Оно создается с cObject (“contentObject”), называемым HMENU. Это расшифровывается как “Hierarchical Menu”. Когда вы инициализируете “HMENU”, вы можете выбрать номер уровня вашего меню. Вы можете таже выбрать, основывается ли оно на HTML-тексте или на gif-изображении.

В первом руководстве мы создавали TEXT-ое меню. Поэтому в данном случае создадим графическое:

Прежде всего, добавьте это(выделено красным) в TypoScript шаблон (в "menu"-секцию):

```
# Page-content
menu.10 = TEMPLATE
menu.10 {
  template = FILE
  template.file = fileadmin/frames/menu.html
  workOnSubpart = DOCUMENT_BODY
  subparts.MENU_ITEMS = HMENU
  subparts.MENU_ITEMS {


  }
}
```

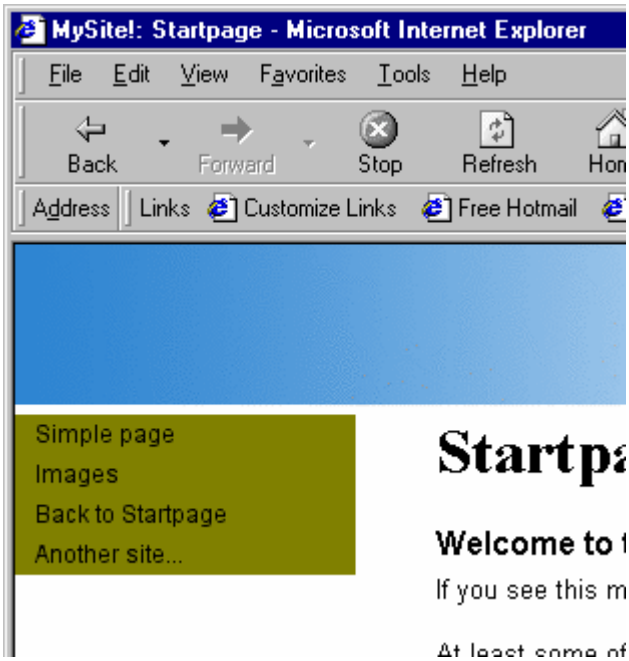
Это создаст четыре пункта меню. Они генерируют себя сами!

Посмотрим в TSRef под "HMENU", вы увидите, что также необходимо определить "menuObj" для получения всего, что выводится.

Модифицируйте шаблон как показано далее (красные линии изменены):

```
# Page-content
menu.10 = TEMPLATE
menu.10 {
  template = FILE
  template.file = fileadmin/frames/menu.html
  workOnSubpart = DOCUMENT_BODY
  subparts.MENU_ITEMS = HMENU
  subparts.MENU_ITEMS.1 = GMENU
  subparts.MENU_ITEMS.1.NO {
    XY = 200,20
    backColor = olive
    10 = TEXT
    10.text.field = title
    10.offset = 10,13
  }
}
```

... и вы увидите это:



Вот что произошло:

Первый "menuObj" HMENU определен как"GMENU" (посмотрите выше код!). GMENU-объект имеет достаточно свойств, чтобы определить свои различные состояния пунктов. Например, состояния "NO", "RO", "ACT"...

"NO" значит "normal", в то время, как "RO" значит "RollOver". Мы увидим эффект всего этого ниже. Но сейчас мы определим дефолтное состояние ("NO"), определив пункт меню установкой GIFBUILDER-объекта (посмотрите выше!) для "NO".

GIFBUILDER-объект очень распространен в Typo3. Он также достаточно мощный, поэтому у вас может возникнуть желание изучить его. Но в любом случае, ключ к какому-либо действию с GIFBUILDER-объектом состоит в определении размера результирующего gif-файла (или PNG-файла, если PNG-опция установлена в порядке создания PNG-файла хотя Typo3 обычно создает GIF-файлы).

Размер установлен в 200x20 точек и цвет фона определен в "olive" (HTML-имя цвета).

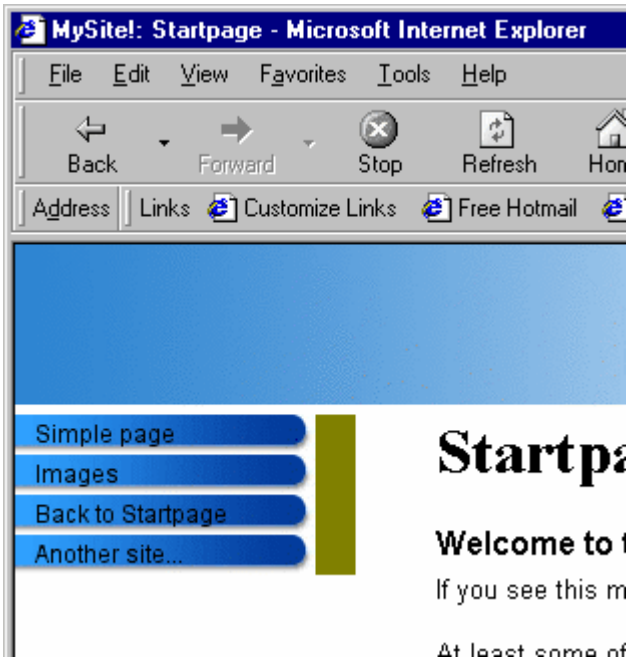
Следующий GIFBUILDER-объект, "TEXT" располагается в позиции 10 в массиве GIFBUILDER-объекта. "TEXT"-объект имеет ".text"-свойство, которое имеет "stdWrap"-свойство, которое позволяет нам "импортировать" поле заглавия текущего пункта меню! Также мы видим смещение текста, который прорисовывается внутри изображения.

Фоновое изображение

Нам наверняка не понравится иметь фоном цвет olive и есть желание вместо этого положить оригинальный gif-file. Это делается очень легко установкой другого GIFBUILDER-объекта, "IMAGE" *перед* "TEXT"-объектом в позиции 10.

Добавьте эти линии:

```
subparts.MENU_ITEMS = HMENU
subparts.MENU_ITEMS.1 = GMENU
subparts.MENU_ITEMS.1.NO {
  XY = 200,20
  backColor = olive
  5 = IMAGE
  5.file = fileadmin/frames/menuback.gif
  10 = TEXT
  10.text.field = title
  10.offset = 10,13
}
```



Все довольно неплохо. Но действительно, мы бы хотели, чтобы gif-файл имел теже самые размеры как и giffile, который мы положили в качестве фона.

Мы *можем* подкорректировать значение "XY = " непосредственно, но другим вариантом будет позволить Typo3 *получить* значение из текущего "IMAGE" объекта! Так модифицируйте ваш шаблон подобно:

```
subparts.MENU_ITEMS = HMENU
subparts.MENU_ITEMS.1 = GMENU
subparts.MENU_ITEMS.1.NO {
  XY = [5.w],[5.h]
  backColor = olive
  5 = IMAGE
  5.file = fileadmin/frames/menuback.gif
  10 = TEXT
  10.text.field = title
  10.offset = 10,13
}
```

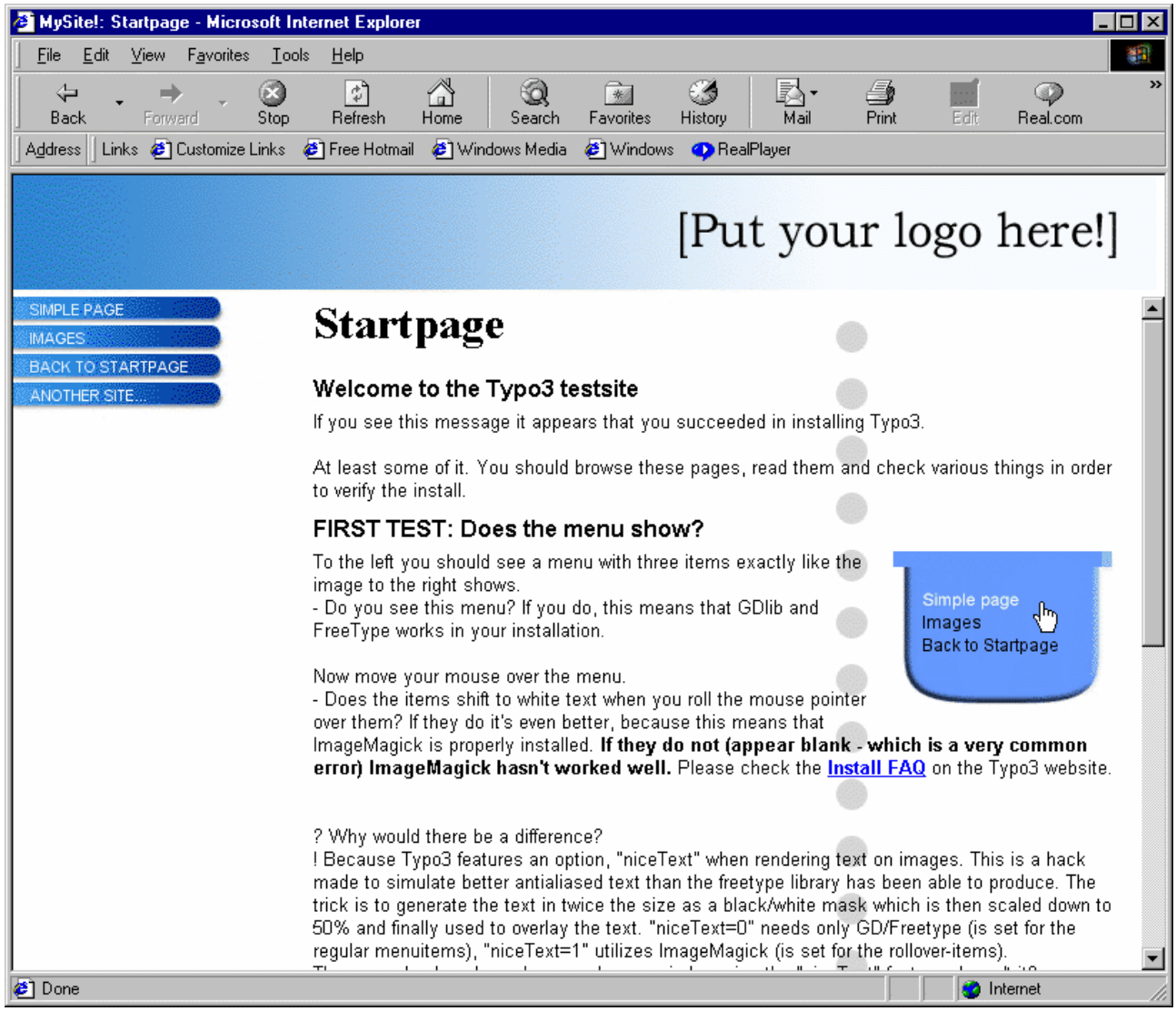
Таким образом Typo3 вставит ширину и высоту GIFBUILDER-объекта 5 (image) как размер!

В конце, мы хотели бы изменить цвет и размер текста, используя другой “fontface”, установить названия менюшек в верхний регистр и немного сместить текст.

Так же важно установить цель GMENU. Это устанавливается к *имени* PAGE-объекта (которое является всегда именем фрейма!) с которым мы хотим связать ссылку на открытие новой страницы.

```
subparts.MENU_ITEMS = HMENU
subparts.MENU_ITEMS.1 = GMENU
subparts.MENU_ITEMS.1.target = page
subparts.MENU_ITEMS.1.NO {
  XY = [5.w],[5.h]
  5 = IMAGE
  5.file = fileadmin/frames/menuback.gif
  10 = TEXT
  10.text.field = title
  10.text.case = upper
  10.offset = 12,12
  10.fontColor = white
  10.fontFile = t3lib/fonts/verdana.ttf
  10.fontSize = 11
}
```

... и вот что мы получим:


MySite!: Startpage - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit Real.com
Address Links Customize Links Free Hotmail Windows Media Windows RealPlayer
[Put your logo here!]
SIMPLE PAGE
IMAGES
BACK TO STARTPAGE
ANOTHER SITE...
Startpage
Welcome to the Typo3 testsite
If you see this message it appears that you succeeded in installing Typo3.
At least some of it. You should browse these pages, read them and check various things in order to verify the install.
FIRST TEST: Does the menu show?
To the left you should see a menu with three items exactly like the image to the right shows.
- Do you see this menu? If you do, this means that GDlib and FreeType works in your installation.
Simple page
Images
Back to Startpage
Now move your mouse over the menu.
- Does the items shift to white text when you roll the mouse pointer over them? If they do it's even better, because this means that ImageMagick is properly installed. If they do not (appear blank - which is a very common error) ImageMagick hasn't worked well. Please check the Install FAQ on the Typo3 website.
? Why would there be a difference?
! Because Typo3 features an option, "niceText" when rendering text on images. This is a hack made to simulate better antialiased text than the freetype library has been able to produce. The trick is to generate the text in twice the size as a black/white mask which is then scaled down to 50% and finally used to overlay the text. "niceText=0" needs only GD/Freetype (is set for the regular menuitems), "niceText=1" utilizes ImageMagick (is set for the rollover-items).
Done
Internet

Без-frames версия

Если вам нравится просматривать сайты старыми броузерами, не поддерживающими фреймы, вы можете легко это сделать.

Попробуйте добавить это внизу вашей шаблонной записи:

```
# ************************
# Defining how the site should look with NO frames
# ************************

noframes = PAGE
noframes.typeNum = 0

# Header code, stylesheet
noframes.headerData.10 = TEMPLATE
noframes.headerData.10 {
  template = FILE
  template.file = fileadmin/frames/content.html
  workOnSubpart = HEADER_CODE
}

# Bodytag
noframes.bodyTag = <BODY bgcolor="#ffffff" background="fileadmin/frames/background.gif"/>
noframes.bodyTagMargins = 0

# Page-content
noframes.10 = TEMPLATE
noframes.10 {
  template = FILE
  template.file = fileadmin/frames/noframes.html
  workOnSubpart = DOCUMENT_BODY
  marks.TOP < top.10
  marks.CONTENT < page.10
  marks.MENU < menu.10
  marks.MENU.subparts.MENU_ITEMS.1.target = _top
}
```

Теперь посмотрим, как это выглядит... Весь сайт выглядит, как и прежде, но теперь использована таблица вместо фреймов! Что произошло?

Вот что произошло. Обратите внимание на синие линии в коде сверху:

Прежде всего, новый PAGE-объект определен ("noframes") и, конечно, он получил позицию дефолтного type-номера (0)! Поэтому целый сайт теперь генерируется в данной манере! Как вы можете видеть в параметре "noframes" PAGE-объект похож на таковой на других страницах. Но верхний контент (header-content) получен из "fileadmin/frames/content.html". В этом случае я решил, что таблицы стилей будет вполне достаточно.

Позже вы увидите, как "bodyTagMargins" установлен в нуль для устранения пустого пространства между контентом и содержанием страницы в верхнем левом углу страницы.

В конце три маркера ###TOP###, ###CONTENT### и ###MENU### в шаблоне заменяются контентом ... оригинального page-объекта. Сделано это *копированием* TEMPLATE-cObject'ов "top.10", "page.10" и "menu.10", определенных выше. (смотрите описание TypoScript по данному синтаксису TypoScript!)

Но одну вещь мы упустили. Теперь мы не работаем с фреймами и ссылки меню поменялись. Перед тем, как мы установим ссылку на “page” перед открытием контента в контент-фрейме. Теперь нам надо добавить к ссылке "_top". Или ничего. Это делается заменой оригинального значения новым данной строкой:

```
  marks.MENU.subparts.MENU_ITEMS.1.target = _top
```

Заметьте, как точно "location" свойства "target" меню связано с “location” из меню PAGE-объекта!!

Работая сразу с фреймами и без.

Теперь сайт оказался без фреймов. Но мы хотим комбинировать их!

Это делается представлением переменной, "noframes", которая может быть установлена из URL сайта. Мы хотели бы реализовать этот сценарий:

index.php?id=xxx	Идем во фрейм-версию
index.php?id=xxx&noframes=1	Идем не во фрейм-версию

Это может быть сделано на основе *условий* в TypoScript.

Если вы положите этот код перед “no-frames” кодом, ваши мечты легко воплотятся:

```
[globalString= noframes=1]

# ************************
# Defining how the site should look with NO frames
# ************************
........


  marks.TOP < top.10
  marks.CONTENT < page.10
  marks.MENU < menu.10
  marks.MENU.subparts.MENU_ITEMS.1.target = _top
}

[global]
```

(Физически мы добавляем [global]-условие внизу, потому что это заставит TypoScript парсер включать что-либо после этого вне зависимости от того, чем это является. Потому что вы вставляете "noframes"-определение в конец записи шаблона, это не имеет никакого значения.)

"index.php?id=1&noframes=1" теперь будет работать. Но одна вещь все же упущена. Когда вы нажимаете на меню, вы увидите, что сайт вернулся опять к фреймовой версии. Это происходит потому, что переменная "noframes" зарегистрирована с Typo3 как "linkVar", что значит, что она передается со всеми линками, генерируемые Typo3.

Поэтому добавьте это:

```
noframes.config.linkVars = noframes
```

Когда вы посмотрите на ссылку из меню, вы заметите, что переменная "&noframes=1" включена, как параметр в ссылку на страницы. При этом фрейм-версия будет работать в любом случае.

Варианты проекта

Вы можете легко заменять документы в данном руководстве на ваши собственные. Вы также имеете достаточно причин для изменения шаблонов и свободного соединения в различных вариантах HTML-шаблона и TypoScript.

Одну вещь имейте в виду, когда разрабатываете ваши HTML-документа для более позднего выполнения с Typo3, и состоит она в том, что все ваши внешние медиа файлы картинки, звуки, stylesheets и тп – необходимо держать в директории “ниже” директории, в которой располагается ваш html-файл. И эта директория будет называться "fileadmin" для того, чтобы Typo3 мог обратиться к ней.

Пример:

Вы создаете новый сайт в папке "new_site" на вашем жестком диске, положите index.html в эту папку, но создайте другую папку, "fileadmin" и внутри нее еще одну папку "images" (или что-то подобное), в которую вы положите все ваши рисунки.

Это создаст файловую структуру подобно:

```
new_site/index.html
new_site/fileadmin/newsite_resources/test.jpg
new_site/fileadmin/newsite_resources/test2.jpg
```

Когда вы уже начинаете работать с Typo3, вы делаете копию "new_site/index.html" в "new_site/fileadmin/newsite_resources/index.html" и копию целой папки "newsite_resources/" в существующую папку "fileadmin/" на Typo3 сервере.

Когда вы используете файл "index.html" как HTML-шаблон (как продемонстрировано в руководстве) все ссылки на внешние файлы будут таким образом не повреждены. Причина этого в том, что "index.html" включается index.php-скриптом, который размещен в корне сайта. Поэтому ссылки на папку "fileadmin/newsite_resources/" корректны.