

Sudo Portal Kullanım Klavuzu

ubuntu-tr

Haziran, 2016

İçindekiler

1	Başlarken	3
1.1	Giriş	3
1.2	Hızlı Başlangıç	3
1.3	Yerelde Çalışma	4
2	İçerik	6
2.1	Dosya Yapısı	6
2.2	Makale İşlemleri	6
2.3	Kategori İşlemleri	7
2.4	Etiket İşlemleri	8
2.5	Yazar İşlemleri	8
2.6	Özel Yapı İşlemleri	9
2.7	Sayfa İşlemleri	10
2.8	Veritabanı İşlemleri	10
3	Markdown	12
3.1	Metin	12
3.2	Başlık	13
3.3	Liste	13
3.4	Tanımlama	13
3.5	Resim	14
3.6	Video	14
3.7	Alıntı	14
3.8	Kod	14
3.9	Tablo	15
4	Özelleştirme	16
4.1	Sistem Yapısı	16
4.2	Tema Seçimi	16
4.3	Şablon Seçimi	17

4.4	Stil Seçimi	18
4.5	Tema Ayarı	18
4.6	Uzantı Ayarları	18
4.7	Kancalama Bölgeleri	18
5	İleri Seviye	20
5.1	YAML	20
5.2	Liquid	20
5.3	Jekyll Değişkenleri	20
5.4	Stil Oluşturma	20
5.5	Şablon Oluşturma	20
5.6	Tema Oluşturma	20
5.7	Uzantı Oluşturma	21
6	Bitirirken	22
6.1	Projeye Katkıda Bulunma	22
6.2	Lisans	22

1 Başlarken

1.1 Giriş

Sudo Portal, aylık yayın yapan Ubuntu Türkiye E-Dergisi SUDO’nun sürekli yayın hayatına geçişi için hazırlanmış web iskeletidir. İçerik yönetim sistemlerinden farklı olarak her değişiklik sonrası tüm sayfalar otomatik oluşturulur ve bu oluşan statik sayfalar ziyaretçilere gösterilir. Sudo Portal, web yayını yapabilmek için işlenen süreci en aza ve basite indirmeyi amaçlamıştır. Bundan dolayı veritabanı gibi karmaşık yapıları kullanmaz.

Web iskeleti ve üzerine eklenmiş verileri statik sayfalara dönüştürmek için jekyll yazılımını kullanır. Dolayısıyla bu iskelet jekyll ile tam uyumlu olacak şekilde geliştirilmiştir. Github Pages, Github üzerinde bulunan jekyll uyumlu sistemleri, jekyll kullanarak otomatik statik sayfalara dönüştürerek yayına alır. Dolayısıyla barındırma gibi işlemlerle uğraşmak zorunda kalınmaz.

Tüm sistem git sürüm kontrol sistemi kullanılarak geliştirilmektedir. Dolayısıyla tüm değişiklikler kayıt altına alınıp yorumlanır. Sorun çıkması durumunda istenilen bir noktaya geri dönüş oldukça kolaydır. Ayrıca bu sistem değişiklikleri dosya bazında değil, satır satır kontrol eder. Sonuç olarak güncelleme yapılacağı zaman bir uyumsuzluk ya da kendi verilerinin üzerine yazılması gibi bir durumun yaşanma olasılığı oldukça düşüktür.

Tüm dosyalar Github üzerinde barındırılır. Github hem git sürüm kontrol sistemini destekler hem de Github Pages servisiyle, jekyll yapılı sistemleri otomatik yayına alır. Üzerinde hata takip sistemi, yorumlama sistemi, sürüm yönetim sistemi gibi oldukça önemli servisleri barındırdığı için tüm proje tek yerden yönetilir.

Görsellik için Semantic UI çatısı varsayılan olarak kullanılır. Her türlü görsel elemanları üzerinde barındıran oldukça güçlü bir çatı olduğu için tasarım yapımı ve yönetimini oldukça kolaylaştırır.

Sistemin güvenlik sorunu yok çünkü dinamik bir yapıya sahip değil. Bu da rahat bir kullanım olanağı sağlar.

Sudo Portal tamamen özgür şekilde topluluk desteği ile geliştirilmektedir. Herkesin bu sisteme katkıda bulunabileceği gibi herkes bu sistemi kendisi için kullanabilir. Bu sistemi kullanıp kendi yayını oluşturmak sadece birkaç tıktan ibarettir. Her şey en basite indirgenmiştir.

1.2 Hızlı Başlangıç

Bu kısımda sitenizi oluşturup yayına alacağız.

- İlk önce [Github’a](#) giriş yapın. Eğer üyeliğiniz yoksa ücretsiz olarak üyelik oluşturun.
- Daha sonra [Github Portal](#) proje sayfasında sağ üstte bulunan “fork” düğmesine tıklayarak çatallayın. Artık sistem kendi kullanıcı hesabınız üzerine aktarıldı.
- Hesabınızda bulunan ve biraz önce çatalladığımız ubuntu-tr.github.io isimli depoyu açın ve “Settings” düğmesine tıklayarak ayarlar sayfasına ulaşın. Burada deponun adını (“Repository name”) *kullaniciAdi.github.io* şeklinde değiştirin.
- Sistemde bulunan bir içerikte değişiklik yaptığımız takdirde Github Pages statik sayfaları oluşturup siteyi yayına alacaktır. Bunun için az önceki depoyu tekrar açarak “README.md” dosyasında değişiklik yapıp kaydedin.
- Birkaç dakika sonra <https://kullaniciAdi.github.io> adresinde siteniz yayına başlamış olacaktır. Sitenizi ziyaret edebilirsiniz.

1.2.1 gh-pages üzerinden yayına alma

Eğer yayını <https://kullaniciAdi.github.io/projeAdi/> adresiyle yapmak istiyorsanız dosyalarınız gh-pages dalı üzerinde bulunmalıdır. Bunun için bir dosyayı açıp (örneğin README.md) küçük değişiklik yapın. Kaydetme düğmesinin hemen üstünde bulunan “Create a new branch for this commit and start a pull request.” kutucuğunu işaretleyip altında açılan alana gh-pages yazın. Kaydettikten sonra siteniz yayına başlayacaktır.

1.2.2 Kendi alan adıyla yayına alma

“Settings” düğmesi ile proje ayarlarına ulaşın. Bu sayfada “Custom domain” alanına yayınlamak istediğiniz alan adını yazın ve kaydedin.

1.3 Yerelde Çalışma

Büyük değişiklikleri Github üzerinde yapmak zordur. Ayrıca orada yapacağınız değişiklikler anında siteye yansıtacağı için yerelde çalışmak daha rahat bir ortam sağlar. Yerelde çalışabilmek için aşağıdaki adımları takip edeceğiz.

- Git ile dosyaları yerele çekme
- Jekyll yazılımını yükleme
- Jekyll ile statik sayfaları oluşturma

1.3.1 Dosyaları yerele çekme

Dosyaları git kullanarak yerele çekeceğiz. Eğer sisteminizde git yüklü değilse yükleyin.

Ubuntu için;

```
1 sudo apt-get install git
```

Daha sonra içinde sistemin barındırılacağı boş bir klasör oluşturup bu klasör üzerinde uçbirimi açıp aşağıdaki komut ile dosyaları buraya çekin.

```
1 git clone https://github.com/ubuntu-tr/ubuntu-tr.github.io.git .
```

Dosyaları indirmek internet bağlantınıza bağlı olarak biraz zaman alacaktır.

1.3.2 Jekyll'i yükleme

Aşağıdaki komut ile Jekyll yazılımını bilgisayarınıza yükleyebilirsiniz.

Ubuntu 14.04 ve sonraki sürümler için;

```
1 sudo apt-get install jekyll
```

1.3.3 Jekyll ile statik sayfaları oluşturma

Bilgisayarınızda gerekli tüm yazılımlar kurulu olduğuna göre geriye yerele çektiğimiz verilerden statik sayfalar oluşturmak kalıyor. Bunun için uçbirimde verilerin olduğu klasöre geldikten sonra aşağıdaki komutu girin.

```
1 jekyll serve
```

Jekyll statik sayfaları oluşturacak ve verilerde değişiklik olup olmadığını sürekli izleyecek. Verilerde her değişiklik yaptığınızda kendisi otomatik statik sayfaları güncelleyecek. Artık yayında olan sitemize bakmamızın vakti geldi. <http://localhost:4000> bağlantısını internet tarayıcınızda açarak sitenize ulaşabilirsiniz. Jekyll ile işiniz bittiğinde uçbirimde ctrl+c tuşlarına basarak çalışmasını durdurun.

1.3.4 Yereldeki değişiklikleri github'a gönderme

Yaptığınız değişiklikleri sunucuya göndermek için aşağıdaki komutları sırasıyla işletebilirsiniz.

```
1 rm -r _site
2 git pull
3 git add .
4 git commit -am "Düzenleme Notu"
5 git push
```

NOT: Bu komutları bir bash dosyası haline getirip lazım olduğunda çalıştırırsanız daha hızlı çalışırsınız.

2 İçerik

2.1 Dosya Yapısı

Jekyll, başında _ işareti olmayan tüm dosyalardan bir statik sayfa üretir. Eğer bir dosyadan statik sayfa üretilmesi istenmiyor veya farklı şekillerde üretilmesi isteniyorsa dosya adının başına _ işareti koyulur.

Bu sistemde dosya yapısı büyük önem taşır. Bir dosyanın nerede olduğuyla nasıl yorumlanacağı anlaşılır. Sistemdeki dosya yapısı şu şekildedir.

2.1.1 İçerik klasörleri

_posts/

Sitede yayınlanacak tüm yazılar/makaleler bu klasör içerisinde bulunur.

_assets/

Makaleler ile ilgili dosyaları barındırır.

_category/

Tüm kategorileri tanımlayan dosyalar bu klasör içerisinde bulunur.

_tag/ Tüm etiketleri tanımlayan dosyalar bu klasör içerisinde bulunur.

_other/author/

Tüm yazarları tanımlayan dosyalar bu klasör içerisinde bulunur.

_other/

Yazar yapısı gibi bir yapı oluşturulmak isteniyorsa bu klasör altında yeni bir klasör oluşturulur.

_root/

Bu dizin altındaki tüm dosyalar direk statik sayfalara dönüştürülür.

2.1.2 Sistem klasörleri

_data/

Sistemin veritabanını oluşturur.

_extension/

Sistemin işleyişine yardımcı olması için bulunan uzantıları içerir.

_theme/layout/

Sistemin görünüşünü belirleyen şablon dosyalarını içerir

_theme/

Sistemin görünüşünü etkileyen uzantıları içerir.

2.2 Makale İşlemleri

2.2.1 Makale oluşturma

Tüm makaleler “_posts” klasörü altında .md uzantısıyla tanımlanmıştır. Klasör altındaki her bir dosya bir makaleyi temsil etmektedir. Örnek bir makale dosyasının içeriği aşağıda verilmiştir.

```
1 ---
2 title: "Squid"
3 date: 2016-04-19 02:11
4 categories: "k2"
5 tags: ["Sudo 50. Sayı", "vekil sunucu", "sunucu", "önbellek"]
6 permalink: "squid"
7 summary: "... özet ..."
8 image: "1.jpg"
```

```
9 thumb: "1.jpg"
10 author: "Çağrı Emer"
11 —
12 Açıklama
```

Tanımlanması gereken alanları açıklarsak;

title Makalenin başlığı belirtilir.

date Makalenin oluşturulma zamanı yazılır.

categories

Makalenin hangi kategori altında yayınlanacağı belirtilir. Kategori sayfasında tanımlanmış code alanı kullanılır.

tags Makalenin hangi etiketler altında yayınlanacağı belirtilir.

permalink

Makalenin sayfasına ulaşmak için gidilecek bağlantı tanımlanır.

summary

Makale hakkında kısa özet yazılır.

image

Makalenin kapak resminin bağlantısı

thumb

Listeleme sırasında kullanılacak makaleye ait resim bağlantısı

author

Yazarın ismi

açıklama

Makalenin içeriği bu bölgede yazılır. Bu yazı **markdown** formatında olmalıdır.

Bu dosya “2016-04-13-squid.md” gibi yani tarih-isim.md ismiyle kaydedilmeli. Bu yazılmış makaleye http://site_adresi/permalink adresiyle ulaşabiliriz.

2.2.2 Makale silme

Aşağıdaki işlemler yapıldığında makale sistemden silinmiş olur.

- _posts/ klasörü içerisindeki ilgili makalenin dosyasını silin.
- images/post/ klasörü içerisindeki ilgili makalenin klasörünü silin.

2.3 Kategori İşlemleri

2.3.1 Kategori oluşturma

Tüm kategoriler “_category” klasörü altında .md uzantısı ile tanımlanmıştır. Klasör altındaki her bir dosya bir kategoriye temsil etmektedir. Örnek bir kategori dosyasının içeriği aşağıda verilmiştir.

```
1 —
2 title: "Haberler"
3 code: k1
4 order: 1
5 color: red
6 —
7 Açıklama
```

Tanımlanması gereken alanları açıklarsak;

title Kategori ismi
code Benzersiz rastgele bir deęer
order Kategorinin hangi sırada olacaęı
color Kategoriyi dięer kategorilerden grsel olarak ayırmak iin bir renk tanımı
aıklama İstenirse kategori ile ilgili aıklama yazılabilir. Bu yazı **markdown** formatında olmalıdır.

Bu dosyayı isim.md ismiyle kaydederek, bu tanımlanmış kategoriye https://site_adresi/isim/ adresiyle ulaşabiliriz.

2.3.2 Kategori silme

Aşağıdaki işlemler yapıldığında kategori sistemden silinmiş olur.

- `_category/` klasörü içerisindeki ilgili kategorinin dosyasını silin.
- Tüm makalelerdeki “categories” alanında belirtilmiş bu kategoriye ait verileri silin.

2.4 Etiket İşlemleri

2.4.1 Etiket oluşturma

Tüm etiketler “_tag” klasörü altında .md uzantısı ile tanımlanmıştır. Klasör altındaki her bir dosya bir etiketi temsil etmektedir. Örnek bir etiket dosyasının içerięi aşağıda verilmiştir.

```
1 —
2 title :   "Güvenlik"
3 —
```

Tanımlanması gereken alanları açıklarsak;

title Etiket ismi

Bu dosyayı `guvenlik.md` ismiyle kaydederek, bu tanımlanmış etikete https://site_adresi/etiket/guvenlik adresiyle ulaşabiliriz.

2.4.2 Etiket silme

Aşağıdaki işlemler yapıldığında etiket sistemden silinmiş olur.

- `_tag/` klasörü içerisindeki ilgili etiketin dosyasını silin.
- Tüm makalelerdeki “tags” alanında belirtilmiş bu etikete ait verileri silin.

2.5 Yazar İşlemleri

Bu işlemler etiket ile ilgili işlemlere çok benzer.

2.5.1 Yazar oluşturma

Tüm yazarlar “_other/author” dizini altında .md uzantısı ile tanımlanmıştır. Dizin altındaki her bir dosya bir yazarı temsil etmektedir. Örnek bir yazar dosyasının içeriği aşağıda verilmiştir.

```
1 ____  
2 title : "Ubuntu TR"  
3 ____
```

Tanımlanması gereken alanları açıklarsak;

title Yazarın ismi

Bu dosyayı ubuntu-tr.md ismiyle kaydederseniz, bu tanımlanmış yazara https://site_adresi/o/ubuntu-tr adresiyle ulaşabiliriz.

2.5.2 Yazar silme

Aşağıdaki işlemler yapıldığında yazar sistemden silinmiş olur.

- _other/author/ klasörü içerisindeki ilgili yazarın dosyasını silin.
- Tüm makalelerdeki “author” alanında belirtilmiş bu yazara ait verileri silin.

2.6 Özel Yapı İşlemleri

Yazar yapısı gibi benzer yapılar oluşturulmak için kullanılır.

2.6.1 Özel yapı oluşturma

Bu tarz tüm yapılar _other klasörü içerisinde bulunur. Yeni yapı oluşturmak için sadece bu klasör altında yeni bir klasör oluşturun. Makalelerde bu özel yapıları kullanmak için _other altında oluşturduğumuz klasörün ismini makalede belirtin. Örneğin: author : "Ubuntu TR"

Tüm özel yapı altındaki verilere aynı şekilde ulaşılır (https://site_adresi/o/özelYapıVeriAdı). Örneğin: https://site_adresi/o/ubuntu-tr

Dikkat edilirse sayfaya ulaşılırken özel yapının klasörü görmezden gelinir. Bu ulaşımın dolaylı tüm özel yapı verilerinin kendi klasörü altında dosyası olsa da asla diğer özel yapı verileri dosyalarıyla aynı ismi taşımamalıdır.

2.6.2 Özel yapı silme

Aşağıdaki işlemler yapıldığında oluşturulmuş özel yapı sistemden silinmiş olur.

- _other/ klasörü içerisindeki ilgili yapının klasörünü silin.
- Tüm makalelerdeki “yapı adı” alanını silin.

2.6.3 Özel yapı verisi

_other klasörü altında oluşturulmuş author klasörü özel yapıdır ve yazar sayfaları oluşturmak için kullanılır. Diğer özel yapı verileri de aynı şekilde oluşturulup / silinir. Detaylı bilgi için o bölüme tekrar bakabilirsiniz.

2.7 Sayfa İşlemleri

Yukarıdaki anlatımlarda hep belirli tür sayfaları oluşturmak için yapıldı. Bir türe bağlı kalmaksızın yapılan sayfa işlemleri burada anlatılacaktır.

2.7.1 Sayfa oluşturma

Tüm türü olmayan sayfalar _root klasörü altında bulunur. Eğer bir sayfa oluşturmak istiyorsanız bu klasörün altına atmanız yeterlidir. Bu klasör altındaki bir dosya siteye aynen yansıyacaktır.

Örneğin a.html isimli bir dosyayı bu klasöre koyduğumuzda bu dosyanın sayfasına https://site_adresi/a.html bağlantısıyla ulaşılır.

b diye klasör açıp a.html dosyasını bu klasör içine koyarsak (_root/b/a) bu dosyanın sayfasına https://site_adresi/b/a.html bağlantısıyla ulaşırız.

2.7.2 Sayfa silme

Bir sayfayı silmek istediğinizde _root klasöründe ilgili sayfanın dosyasını/klasörünü silin.

2.8 Veritabanı İşlemleri

Uzantılar, sürekli değişebilen veriler barındırabilir. Bu veriler _data klasörü içerisinde yaml ya da json formatında dosya olarak bulunur. Sistemde varsayılan olarak 3 uzantı, 3 veri dosyası kullanır. Bu bölümde bu dosyaları inceleyeceğiz.

2.8.1 En çok okunanlar / mostRead.yml

MostRead uzantısının kullandığı veri dosyasıdır. İçerisinde her kategori ve ana sayfa için seçilmiş makaleler tanımlanmıştır. En çok okunanlar kısmında listelenen makaleler burada yazılmış makalelerdir. İçeriğinin bir kısmı şu şekildedir.

```
1 - thumb: heartbleed.jpg
2   title: "Heartbleed"
3   url: /heartbleed
4   category: home
5
6 - thumb: 1.jpg
7   title: "Eski Bir Bilgisayar Bir Dosya Paylaşım Sunucusuna Nasıl Çevrilir?"
8   url: /samba-server
9   category: home
10
11 - thumb: heartbleed.jpg
12   title: "Heartbleed"
13   url: /heartbleed
14   category: k1
```

Her makale başında - işareti ile tanımlanmış ve yanında makalenin özellikleri belirtilmiş. Burada önemli kısım category alanıdır. Bu alanda belirtilen değere göre farklı sayfalarda farklı listeleme yapılır. k1, k2 gibi kategori kodları içeriyorsa o kategoriye aittir veya home değeri içeriyorsa kategori sayfaları dışındaki sayfalarda listelenecek demektir.

2.8.2 Duyurular / news.yml

News uzantısının kullandığı veri dosyasıdır. İçerisinde duyuru için listelenecek makaleler tanımlanmıştır. İçeriğinin bir kısmı şu şekildedir.

```
1 - thumb: 1.jpeg
2   title: "MongoDB 3.2 yayınlandı"
3   url: /mongodb
4
5 - thumb: heartbleed.jpg
6   title: "İnternet dünyasının %70'i tehlike altında"
7   url: /heartbleed
```

2.8.3 Slider / slider.yml

Slider uzantısının kullandığı veri dosyasıdır. İçerisinde slider için listelenecek makaleler tanımlanmıştır. İçeriğinin bir kısmı şu şekildedir.

```
1 - image: 2.jpg
2   url: '#'
3 - image: 1.png
4   url: '#'
```

3 Markdown

3.1 Metin

3.1.1 Kalın, eğik veya satırıçi kod yazma

Kullanım

Bu **komutlar** içerisinde ‘apt-get’ gibi ****çok önemli**** bilgiler vardır.

Sonuç

Bu *komutlar* içerisinde apt–get gibi **çok önemli** bilgiler vardır.

3.1.2 Paragraf

NOT

Paragraf için arada bir satır bırakılmalıdır.

Kullanım

Bu bir paragraf
Bu da başka bir paragraf

Sonuç

Bu bir paragraf
Bu da başka bir paragraf

3.1.3 Bağlantı verme

Kullanım

<https://www.google.com.tr> bağlantısından veya [Google](https://www.google.com.tr) üzerinden ulaşabilirsiniz.

Sonuç

<https://www.google.com.tr> bağlantısından veya [Google](#) üzerinden ulaşabilirsiniz.

3.1.4 Dipnot

Kullanım

Bu yazı dipnot içeriyor^[1].
[1]: Bu dipnotun açıklaması.

Sonuç

Bu yazı dipnot içeriyor¹.

3.2 Başlık

NOT

Bir başlık yazılmadan önce mutlaka üstünde boş bir satır bırakılmalı.

Kullanım

```
## Başlık 2
### Başlık 3
#### Başlık 4
```

3.3 Liste

NOT

Listelemeye başlamadan önce ve listeledikten sonra boş satır bırakın.

Kullanım

```
1. Bir yazı
2. Başka bir yazı
3. Bu yazılar çok başka
```

Sonuç

```
1. Bir yazı
2. Başka bir yazı
3. Bu yazılar çok başka
```

Kullanım

```
- Bir yazı
- Başka bir yazı
- Bu yazılar çok başka
```

Sonuç

```
• Bir yazı
• Başka bir yazı
• Bu yazılar çok başka
```

3.4 Tanımlama

Kullanım

```
Tanımlanacak İsim
: Buraya tanım yazılacak
```

Sonuç

```
Tanımlanacak İsim
Buraya tanım yazılacak
```

3.5 Resim

NOT

Resmin yanına başka bir şey gelmesini istemiyorsanız resimden önce ve sonra boş satır bırakın.

Kullanım

![Logo](https://ubuntu-tr.github.io/theme/ubuntu-tr/images/cc-by-sa2.png)

Sonuç



3.6 Video

Kullanım

<https://www.youtube.com/embed/W-WihPoEbR4>

Sonuç

<https://www.youtube.com/embed/W-WihPoEbR4>

3.7 Alıntı

NOT

Alıntıdan önce ve sonra boş satır bırakın.

Kullanım

> Bu bir alıntı ve
satır başı yaptım ama
> bu şekilde de satır başı oluyor.

Sonuç

Bu bir alıntı ve
satır başı yaptım ama
bu şekilde de satır başı oluyor.

3.8 Kod

NOT

Kodların başında ve bitiminde boş satır bırakın.

Kullanım

```
““php
<?php
echo “Hello World!”;
?>
““
```

Sonuç

```
1 <?php
2     echo "Hello World!";
3 ?>
```

3.9 Tablo

Tablo oluşturmak için araya | işareti koymak yeterlidir.

NOT

| işaretinin tablo oluşturması istenmiyorsa \| şeklinde kullanılmalıdır.

Kullanım

Hücre 1 Hücre 2 Hücre 3
:— — :—
4 En Geniş Hücre 5 Hücre 6
7 8 9

Sonuç

Hücre 1	Hücre 2	Hücre 3
4	En Geniş Hücre 5	Hücre 6
7	8	9

4 Özelleştirme

4.1 Sistem Yapısı

Özelleştirmeye başlamadan önce sistemi tanımak gerekir. Sistemin parçaları yaklaşık olarak aşağıdaki gibidir.

- **Uzantılar**

Sistemin kolay yönetilebilmesi için sistem birçok parçaya ayrılmıştır. En temel işlemler çekirdekte bulunup diğer parçalar çekirdeğe dışarıdan bağlanır. Bu parçaları uzantı olarak adlandırıyoruz. Tema ile ilgili uzantılar ilgili tema dizininde (`_theme/ubuntu-tr`) bulunurken diğer uzantılar ise `_extension` klasöründe bulunur.

- Kanca

Uzantının sistemin neresine bağlanacağı belirtilir. (“`index: sidebar`” gibi)

- Ayar

Uzantı ile ilgili ayarları içerir. (“`disqus_id: deneme`” gibi)

- Veritabanı

Sürekli değişmesi durumuna sahip verilerin tutulduğu yerdir. bu veriler `_data` klasöründe bulunur.

- **Tema**

Sitenin görünümü belirleyen dosyalar burada bulunur.

- Şablon

Sitenin genel hatlarını belirleyen dosyalardır.

- Stil

Şablonlara görünüm kazandıran dosyalardır. Uzantı olarak tanımlanmışlardır. Tema dizininde `class` klasöründe bulunur.

- **Sayfa Türleri**

Tüm sayfalar bir türe sahiptir ve bu tür üzerinden topluca yönetilirler. Uzantılar farklı türdeki sayfalarda farklı yerlere dahil edilebilir.

- Index

Sadece anasayfa bu türdedir.

- Post

Tüm makale sayfaları bu türdedir.

- Category

Tüm kategori sayfaları bu türdedir.

- Tag

Tüm etiket sayfaları bu türdedir.

- Other

Yazar gibi tüm özel yapı sayfaları bu türdedir.

İleri seviye bölümünde daha ayrıntılı incelenecektir.

4.2 Tema Seçimi

Temalar, sistemin görünümünü baştan sona etkileyen en önemli bileşenlerdir. Sistemde birden fazla tema bulunuyorsa bu temalardan hangisini kullanacağınızı seçebilirsiniz. Öncelikle sistemde hangi temaların yüklü olduğunu öğrenmek için `_theme` klasörünün içindeki klasör isimlerine bakın. Bu dizindeki her bir klasör bir temayı temsil eder. Hangi temayı kullanacağınıza karar verdikten sonra bu seçimimizi sisteme bildirmemiz gerekiyor. Bunun için `_config.yml` dosyasını açın. Bu dosyanın içeriğinin üst kısımları aşağıdaki gibidir.

```
1 # Site settings
2 theme: ubuntu-tr
3 themeDir: _theme/ubuntu-tr
4 listDir: _theme/ubuntu-tr/_layouts/list
5 blockDir: _theme/ubuntu-tr/_layouts/block
6 layouts_dir: ../_theme/ubuntu-tr/_layouts
7 includes_dir: .
```

Burada hangi temayı kullanmak istiyorsak o temanın adını ubuntu-tr yazan yerlerde değiştirmemiz yeterlidir.

NOT

Eğer sistemi jekyll serve komutu ile çalıştırmışsanız ve çalışır durumdaysa, değişikliğin etkin olabilmesi için durdurup yeniden çalıştırmanız gerekir.

4.3 Şablon Seçimi

Her temanın içerisinde sitenin genel hatlarını belirleyen şablon dosyaları bulunur. Bu şablon dosyaları, üst kısım - ana gövde - yan sütun - alt kısım gibi yapısal ayrımları yapar. Eğer 2 yan sütunlu bir sistem kullanmak veya yan sütun olmayan bir sistem kullanmak gibi seçimler yapmak isterseniz şablonu değiştirmeniz yeterlidir.

Şablon dosyaları ilgili tema klasörünün içindeki _layout klasörü içerisinde bulunur. Bu klasördeki dosya isimlerine bakarak hangisini kullanmak istediğinize karar verdikten sonra bu seçimimizi sisteme bildirmemiz gerekiyor. Bunun için _config.yml dosyasını açın. Bu dosyanın aşağıdaki kısımlarında layout alanını değiştirerek tanımlama işlemini yapmış olursunuz. Örneğin ubuntu-tr temasında varsayılan şablon layout121'dir.

NOT

Her sayfa türü için bir şablon seçmeniz gerekmektedir. Sayfa türü layout alanının üzerindeki type alanında belirtilmiştir.

```
1 defaults:
2   - scope:
3     path: ""
4     type: category
5   values:
6     type: categories
7     layout: layout121
8   - scope:
9     path: ""
10    type: tag
11  values:
12    type: tags
13    layout: layout121
14  - scope:
15    path: ""
16    type: posts
17  values:
18    type: posts
19    layout: layout121
20  - scope:
21    path: ""
22    type: other
23  values:
24    type: others
25    layout: layout121
```

NOT

Eğer sistemi jekyll serve komutu ile çalıştırmışsanız ve çalışır durumdaysa, değişikliğin etkin olabilmesi için durdurup yeniden çalıştırmanız gerekir.

4.4 Stil Seçimi

Şablon dosyaları sayfa yapısını belirlerken stil dosyaları ise bu yapıları görsellik katar. Sistemde bulunan hazır stiller arasında seçim yaparak sitenin görünümünü değiştirebilirsiniz. Öncelikle sistemde tanımlanmış stil dosyaları `_theme/temaAdı/class` dizini içerisinde bulunur. Class dosyaları tüm görünüme etki edebileceği gibi sadece küçük bir bölgenin görünümünden de sorumlu olabilir. Kullanmak istemediğiniz class dosyalarının isminin başına “_” işaretini eklerseniz pasif hale getirmiş olursunuz. Tasarımın düzgün gözükmesi için her şeyin eksiksiz tanımlanması gerektiğinden class dosyaları beraber kullanılabilir. Dosyalardaki sıralamaya göre bir dosya diğer dosyanın üzerine yazabilir. Sıralama dosya içerisindeki “order: sıraNO” şeklindeki alan ile belirtilmiştir.

4.5 Tema Ayarı

Temada kullanılan uzantıların ayarlarını yine o uzantılara ait dosyaların içerisinde yapabilirsiniz. Buna ilave olarak görünümle ilgili ayarlar class klasörü içerisinde yapılır. Tema seçimi ise ana dizindeki `_config.yml` dosyasından yapılır.

4.6 Uzantı Ayarları

Sistemde yüklü tema ile ilgili uzantılar `_theme/temaAdı/` dizininde ve geri kalan tüm uzantılar `_extension` klasöründe bulunur. Bu uzantılar ile ilgili ayarlar kendi dosyalarının başında bulunur. Bu bölümde uzantı ayarları kullanılarak uzantılar özelleştirilecektir.

Örnek olarak ubuntu indirme linklerinin bulunduğu kutucuğu oluşturan uzantıyı inceleyelim. Bu uzantı tek dosyadan oluşmakta ve `_extension/downloads.html` adresinde bulunmaktadır. Eğer dosyayı pasif yapmak isteseydik `downloads.html` ismini `_downloads.html` olarak değiştirebilirdik. Dosyanın içeriğine bakarsak;

```
1 order: 6
2 index: sidebar
3
4 header: "Ubuntu İndirme Bağlantısı"
```

Html çıktılar

şeklinde bir içerikle karşılaşırız. Yukarı kısım uzantının ayar bölgesini barındırıyorve ondan sonraki kısım ise html kodlarından oluşuyor. Bu yapıyı biraz daha incelersek;

- **index: sidebar** ile sadece index türündeki sayfalarda sidebar kısmına eklenmesi söylenmiş
- **order: 6** kaydıyla da eklendiği alandaki sırası belirtilmiştir.
- **header: “Ubuntu İndirme Bağlantısı”** şeklinde bir tanımla da oluşacak kutucuğa başlık girdirilmiş.

Html çıktıları da index türündeki sayfaların sidebar kısmında 6. sırada “Ubuntu İndirme Bağlantısı” başlıklı kutucuğun içerisine eklenecektir.

4.7 Kancalama Bölgeleri

Kancalama yerleri şablona göre değişebilirken genellikle aşağıdaki gibidir.

Gösterim Alanları

- Top
- Main
- Sidebar

- Bottom

Css ve javascript ekleme yerleri

- Head (satır içi css için)
- Css
- Js

Diğer

- Meta

5 İleri Seviye

Sistemde veriler yaml formatında tutuluyor. Tema ve eklentilerde dinamik işlem yapma işlemini yani kod yazma olayını liquid sistemi ile yapıyoruz. Dolayısıyla bu bölümü anlamak için bu iki sisteme hakim olmak gerekiyor.

5.1 YAML

Buralar doldurulacak

5.2 Liquid

Buralar doldurulacak

5.3 Jekyll Değişkenleri

Buralar doldurulacak

5.4 Stil Oluşturma

Stil dosyaları, şablonda belirtilmiş yapılara `class` değeri vererek görselliğin oluşmasını sağlar. Ubuntu-tr temasında `_theme/temaAdı/class/class1.yml` dosyasında gereken tüm `class` tanımlamaları yapılmıştır. Bu dosya baz alınarak tamamen ya da kısmen `class` tanımlaması yapılarak yeni bir stil oluşturulabilir. Ubuntu-tr teması varsayılan olarak Semantic UI arayüz tasarım çatısı kullanır. Class değeri ya bu çatı baz alınarak ya da ayrı bir css dosyasında tanımlanmış değerlere göre verilir. Hangi alanın nereye etki ettiği `_theme/temaAdı/_layout` dizinindeki dosyalar incelenerek öğrenilebilir.

Örnek bir stil dosyası;

```
1 grid: "ui grid equal width stackable celled"
2 row1column: "column clearPadding"
3 row2column1: "column white"
4 row2column2: "column five wide computer only lightGrey"
```

5.5 Şablon Oluşturma

Şablon dosyaları `_theme/temaAdı/_layout` dizini içerisinde bulunur. Sitenin genelini etkileyen şablon dosyaları bu dizin altında bulunurken, listelemeyi etkileyen şablon dosyası `list` klasörü içinde bulunur. `layout121.html` dosyası baz alınarak farklı şablon dosyaları hazırlanabilir.

5.6 Tema Oluşturma

Şablonlar, stil dosyaları ve css, js gibi dosyalar ile tema oluşmuş olur. Bu dosyalar `_theme` klasöründe temanın adına sahip klasör içinde barındırılırken, temanın ayar dosyaları `_data` klasörü içerisinde yine temanın adına sahip klasör içinde bulunur.

5.7 Uzantı Oluşturma

Uzantı dosyaları `_extension` klasörü ve `_theme/temaAdı/` klasörü içerisinde bulunur. Ayrıca uzantının kullandığı veriler `_data` klasörü içerisinde dosya olarak tutulurlar.

Uzantı oluşturmak oldukça kolaydır. Eğer uzantı tema ile ilgiliyse `_theme/temaAdı/` dizininde, değilse `_extension` dizininde yeni bir klasör oluşturularak uzantıyla ilgili dosyalar içine atılır. Geriye uzantının hangi aşamada dahil edileceği ve sonradan kontrolün sağlanması için nasıl yapılandırılması gerektiği kalıyor.

Diyelimki uzantımız bir slider olsun. Html, css ve js dosyasından oluşsun. Bu slider sadece kategori sayfalarında ve ana sayfada görünmesini isteyelim. Bunun için öncelikle her dosyanın içeriğinin başına

```
1 ____
2 ____
```

eklenir ve bu alanlar içerisinde bu tanımlar yapılır. Sliderın üst bölgede görüneceğini düşünürsek html dosyasının başı şu şekilde görünür;

```
1 ____
2 index: top
3 category: top
4 ____
```

Aynı şekilde javascript dosyasının içeriği

```
1 ____
2 index: js
3 category: js
4 ____
```

olur. Css dosyalarında ise küçük bir ayrıntı vardır. Eğer sayfa yüklendiğinde bu uzantı ekranın görünümünün içinde kalıyorsa `“...: head”`, kalmıyorsa `“...: css”` şeklinde tanımlanmalıdır. Buradaki amaç sayfa yüklenme hızını olabildiğince arttırmak için ilk açılışta sayfada olmayan uzantıların sonradan yüklenmesini sağlamaktır. Slider uzantısını üst tarafa koyduğumuzdan ekranda görüneceği için içeriği şöyle olacaktır;

```
1 ____
2 index: head
3 category: head
4 ____
```

Bu yapıları ekledikten sonra uzantımız hazırdır. Eğer uzantının diğer uzantılar arasındaki sıralaması önemliyse `“order: 3”` gibi bir tanım yapılabilir.

Uzantı eklendikten sonra kolay kontrol edilebilmesi de önemlidir. Bunun için değiştirilebilen yerleri de yukarıdaki alanda belirtmeliyiz. Örneğin sliderın 10 saniyede bir otomatik değiştiğini düşünün. Bu 10 saniyelik kısmı kodlardan ayırıp yukarı alana taşıyabiliriz. Bunun için yukarı kısma `“time: 10”` gibi bir şey yazarsak kod alanında da 10 yazan yeri `{{ page.time }}` şeklinde değiştirmeliyiz. Bu şekilde önemli yerleri koddan ayırırız.

6 Bitirirken

6.1 Projeye Katkıda Bulunma

Bu projeye çeşitli yollarla katkıda bulunabilirsiniz.

- Bir hata ile karşılaştığınızda [Issues / New issue](#) sayfasından hata kaydı oluşturun.
- [Issues](#) sayfasından açılmış hata kayıtlarına bakabilir ve bilgi sahibi olduğunuz konularda cevap yazabilirsiniz. Ayrıca bazı kararların alınması için açılan tartışma konularına katılabilirsiniz.
- [Makale dizininde](#) bulunan yazıları düzenleyebilirsiniz.
- Bilginiz varsa kodlarda düzenleme yapabilirsiniz.

6.2 Lisans

Bu proje içerisindeki makaleler [CC BY-SA](#) lisansı altındadır. Proje içerisindeki kodlar ise [MIT Lisansı](#) altındadır.

Bu yazılım hiçbir teminat olmaksızın sunulmuştur. Kullanımdan dolayı oluşacak zararlardan kesinlikle yazılım geliştiricileri sorumlu tutulamaz.

<http://ozgurlisanslar.org.tr/mit/>

<http://ozgurlisanslar.org.tr/creative-commons/attribution-sharealike-cc-by-sa/>