Yazılımların Paketlenip Yayımlanması

bugra9

Kasım, 2016

İçindekiler

1 Giriş

Bir yazılımın kolay yüklenip güncellenebilmesi için paketlenip ilgili depolara gönderilmesi oldukça önem taşır. Yazılım ne kadar kolay yüklenir ve ne kadar güven hissi verirse kullanımı da o kadar fazla olacaktır. Bu yazıda güven duygusunu kaybetmeden yazılımların paketlenmesinden bahsedilecektir.

Bir yazılımın açık kaynaklı olması ve kodlarının herkes tarafından incelenebilmesi o yazılımın tehlikeli olabilme olasılığını oldukça düşürür. Diğer yandan açık kaynaklı bir yazılımın yüklenmesi ise oldukça zor bir iştir. Kolay yüklenmesi için yazılımı derleyip paketlersek, kodlar görünür olmayacağı için bu sefer de güven duygusunu kaybettirecektir.

Bu sorunu aşmak için güvenilen 3. bir tarafın/sistemin kodları alıp derleyip paketlemesi gerekmektedir. Bu şekilde olunca o paketin o kaynak kod kullanılarak oluşturulduğundan ve araya bir şey eklenmediğinden emin olmuş oluruz. Bu 3. taraf Ubuntu tarafında Launchpad.net ve Arch Linux tarafında makepkg yazılımı oluyor.

Arch Linux dağıtımında uygulamamızın kolay yüklenmesi için yükleme talimatlarını içeren bir dosya hazırlarız ve bunu aur kullanıcı deposuna göndeririz. yaourt yazılımı ile bu uygulama yüklenmek istendiğinde bu talimatlara uyularak makepkg tarafından uygulama derlenir, paketlenir ve sonrasında pacman tarafından sisteme yüklenir.

Ubuntu tarafında bu sistemin daha gelişmiş hali bulunur. Derleme ve paketleme olayı Ubuntu dağıtımının kendi sunucularında olurken kullanıcıya sadece bu hazır paketi sistemine yüklemek kalır.

Bu yazıda git sürüm kontrol sistemi kullanılarak geliştirilen bir yazılımın nasıl otomatik paketleneceği anlatılmıştır. Paketleme sistemlerinin ortak özelliği makefile dosyasını kullanarak yükleme yapmalarıdır dolayısıyla güzel hazırlanmış bir makefile dosyası olduktan sonra yazılımın herhangi bir sisteme göre paketlenmesi oldukça kolaydır. Makefile dosyasının nasıl hazırlanacağı http://www.belgeler.org/howto/makefile-nasil-kullanimi.html sayfasında anlatıldığı için burada yazmaya gerek duyulmamıştır. Örnek bir makefile dosyasına https://github.com/bugra9/BootableUSB/blob/master/Makefile bağlantısından ulaşabilirsiniz.

2 Deb paketi için debian dizininin ayarlanması

Yazdığınız yazılımın .deb olarak paketlenebilmesi için bazı bazı dosyalara ihtiyacı vardır. Bu dosyalar debian dizini altında bulunmalıdır. Bu bölümde bu dosyaları hazırlanması gösterilecektir. Anlatım olabildiğince basit ve temel düzeydedir. Daha ayrıntılı bilgiye https://www.debian.org/doc/manuals/maint-guide/ sayfasından ulaşabilirsiniz.

Yazılımın paketlenmesi için control, copyright, changelog, rules olmak üzere olmazsa olmaz dört dosyaya ihtiyacımız bulunuyor. Bu dosyaları tek tek hazırlayalım. Yardımcı olması amacıyla https://github.com/bugra9/BootableUSB/tree/master/debian sayfasından canlı örneğini inceleyebilirsiniz.

2.1 debian/control

```
Source: bootableusb
Section: utils
Priority: optional
Maintainer: bugra9 <***@gmail.com>
Build-Depends: debhelper (>=9)
Standards-Version: 3.9.8
Homepage: https://github.com/bugra9/BootableUSB

Package: bootableusb
Architecture: all
Depends: ${shlibs:Depends}, ${misc:Depends}
Description: Create bootable usb drives
 BootableUSB allows you to create bootable USB drives for Windows and Linux distributions.
```

- **Source:** Yazılımın kaynak paketinin adı
- **Section:** İşlevine göre yazılımın bulunduğu kategori. http://packages.ubuntu.com/yakkety/ bağlantısından tüm kategorileri açıklamalarıyla birlikte görebilirsiniz. Buraya yazacağınız değer ilgili kategorinin bağlantısında geçtiği gibi olmalı. Örneğin "Yardımcı Araçlar" kategorisinin bağlantısı http://packages.ubuntu.com/yakkety/utils/ şeklindedir ve "Section:" alanına "utils" yazmalıyız. Buraya yanlış bir değer verdiğinizde yazılım paketlenir ama ppa'ya eklenme aşamasında reddedilir.
- **Maintainer:** Paketlemeyi hazırlayan kişinin ismi ve eposta adresi
- **Build-Depends:** Yazılım derlenirken gerekli olan paketler buraya yazılabilir. Bilmiyorsanız dokunmanıza gerek yok.
- **Homepage:** Yazılımın web sayfası
- **Package:** Yazılımın paket adı
- **Depends:** Yazılımın çalışırken hangi paketlere ihtiyacı olduğu burada belirtilir. Yüklenme esnasında burada yazan paketler de yüklenecektir. Örnek bir kullanım: "libc6, libhunspell-1.3-0 | libhunspell-1.4-0, foo (>= 1.2), libbar1 (= 1.3.4)"
- **Description:** Yanına kısa bir açıklama yazılırken altına uzun bir açıklama yapılabilir.

2.2 debian/copyright

Bu dosyada yazılım ile ilgili lisans bilgileri bulunur. Örnek bir dosya içeriği

```
Format: https://www.debian.org/doc/packaging-manuals/copyright-format/1.0/
Upstream-Name: Yazılımın Adı
Source: Yazılımın kaynak kodlarının bulunduğu adres

Files: *
Copyright: Yıl İsim <Eposta Bağlantısı>
License: Lisans Adı

Files: debian/*
```

```
Copyright: Yıl Paketleyenin İsmi <Eposta Bağlantısı>
License: Lisans Adı

...

License: Lisans Adı
 Lisans içeriği
```

https://github.com/bugra9/BootableUSB/blob/master/debian/copyright bağlantısından tüm dosyaları GPLv3 ile lisanslanmış dosyanın içeriğini görebilirsiniz.

2.3 debian/changelog

Bu dosyaya kaynak kodlar üzerinde yaptığınız değişiklik notları yazılır. Örneğin;

```
Paket Adı (Sürümü) unstable; urgency=medium

  * Initial release

 -- İsim <Eposta Adresi>  Mon, 07 Nov 2016 21:02:38 +0200
```

https://github.com/bugra9/BootableUSB/blob/master/debian/changelog bağlantısından canlı örneğini görebilirsiniz.

Kodları github üzerinden alacağımız için değişiklik notları da github üzerinden çekilecektir. Dolayısıyla bu şekilde bir başlangıç yapıp bir daha dokunmanıza gerek yoktur.

2.4 debian/rules

Yazılımın nasıl derlenip yükleneceği bu dosyada yazılır. Yazılımı siz geliştirdiyseniz ve doğru bir MAKEFILE dosyasına sahipseniz bu dosyayı aşağıdaki şekilde bırakabilirsiniz.

```
#!/usr/bin/make -f

%:
	dh $@
```

2.5 debian/compat

```
9
```

2.6 debian/source/format

```
3.0 (quilt)
```

2.7 debian/patches/*

Eğer dosya paketlenirken kaynak koda yama yapmak isterseniz yapılacak değişiklikleri debian/patches dizini altında dosya oluşturup oraya yazabilirsiniz.

3 Launchpad üzerinden .deb paketi oluşturup yayımlanması

İzlenilecek adımlar şu şekilde olacaktır.

- PPA oluşturma
- PPA'nın kullanılabilmesi için OpenPGP anahtarı oluşturma
- Yazılım için proje sayfası oluşturma
- Proje sayfasında github ile senkronize olacak dal oluşturma
- Bu dalın derlenip .deb paketi haline getirilmesi ve açtığımız ppa üzerinde yayınlanması için "recipe" oluşturma

(R) Launchpad üzerinde giriş yapıp profil sayfasına gittiğimizde resimdeki gibi bir sayfayla karşılaşırız.

3.1 PPA Oluşturma

Profil sayfamızda aşağı sol bölgede "Personal package archives" başlığına sahip alanın içerisindeki "Create a new PPA" düğmesi ile ppa oluşturulacak sayfaya gidilir. (R) Bu sayfada url kısmına ppa adını yazarız. Daha sonra ppa:kullanıcıAdı/ppaAdı şekline dönüşecektir. "Display Name" kısmına da görünecek ppa adını yazdıktan sonra koşulları kabul edip "Activate" düğmesi ile ppa'yı aktif hale getiririz.

Artık bu ppa'yı kullanıcılar,

```
sudo add-apt-repository ppa:kullanıcıAdı/ppaAdı
```

komutuyla ekleyebilecekler. Ama OpenPGP anahtarı oluşturmadığımız için hata alacaklardır.

3.2 PPA'nın kullanılabilmesi için OpenPGP anahtarı oluşturma

Bilmeyenler için görsel olarak anlatılacaktır. "Parolalar ve Anahtarlar" uygulaması açılır. Eğer sizde yüklü değilse,

```
sudo apt install seahorse
```

komutuyla yükleyebilirsiniz.

(R) Bu uygulamada yeşil arkaplana sahip düğmesine tıklayıp PGP anahtarını seçin. (R) Bir sonraki adımda isim ve eposta adresinizi girin. İsim alanına istediğinizi yazabilirsiniz ama eposta adresinizi daha sonra doğrulamak isteyecektir. (R) Oluştur düğmesine bastıktan sonra sizden bu anahtar için parola belirlemenizi isteyecek. Parolayı yazdıktan sonra anahtar eklenir. Sol menüden "GnuPG anahtarları" seçeneğine tıklayın ve anahtarınızın listelenmesini bekleyin. Anahtarın eklenip listede gözükmesi biraz zaman alır. Anahtar listelendikten sonra üzerine tıklayıp özellikler deyin. Açılan ekranda "Anahtar ID" değeri ile "Ayrıntılar" sekmesindeki "Parmak İzi" değerini not alın ve yazılımı kapatın.

```
gpg --keyserver hkp://keyserver.ubuntu.com --send-key notAldığımızAnahtarID
```

komutuyla oluşturduğumuz anahtar bilgilerini ubuntu anahtar sunucularına yollayalım. (notAldığımızAnahtarID kısmını kendine göre değiştirmeyi unutma)

Şimdi Launchpad profil sayfamızdan bu anahtarı eklemek için "OpenPGP keys:" alanının yanındaki kaleme tıklayın. Tekrar oturum açmanızı isteyecektir. (R) Açtıktan sonra not aldığımız parmak izini "Fingerprint:" kutucuğuna girip "Import Key" düğmesine tıklayın. Eposta adresinizi doğrulamak için size şifreli bir bağlantı yollayıp bu adrese gitmenizi isteyecek. (R) Epostanıza gelen bu iletide, resimde gösterildiği şekilde şifreli alanın tamamını ev dizininizde yeni bir dosyanın içine kaydedin ve

```
gpg --decrypt dosya.txt
```

komutuyla bu şifreli iletiyi çözün. Karşınıza çıkan bağlantıyı tarayıcınızla ziyaret ettiğinizde onaylanmış olacaktır.

3.3 Yazılım için proje sayfası oluşturma

https://launchpad.net/projects/+new bağlantısından projeyi oluşturacağınız sayfaya gidin. Burada proje ismi ile projeniz hakkında kısa bir yazı yazarak devam edin. Sonraki sayfada lisans seçerek kaydı tamamlayın. Proje sayfanız oluşmuş olacaktır.

3.4 Proje sayfasında github ile senkronize olacak dal oluşturma

Proje sayfasını açtıktan sonra "code" sekmesini açın. (R) Orada en altta "Configure Code" bağlantısına tıklayarak dalımızı ayarlayacağımız sayfaya gidin. "Version control system" olarak "Bazaar" seçili olarak kalsın. "Link or import an existing branch" alanında "Import a branch hosted somewhere else" seçeneğini işaretleyin. "Branch name:" kısmına "master" "Branch URL:" kısmına ise yazılımınızın bulunduğu git bağlantısını yazın. (R) Eğer Github kullanıyorsanız projenin ana sayfasında "Clone or download" yazan yeşil düğmeye basarak öğrenebilirsiniz. Bağlantı yazma yerinin altında bir grup seçeneklerden ise "Git" yazanı seçerek "Update" düğmesi ile ayarları kaydedin. Artık tüm işlem tamamlanmış olup en yakın zamanda git bağlantısından kodlar launchpad üzerine alınacaktır.

(S) (R) İsterseniz otomatik git bağlantısının alınması işlemini elinizle de tetikletebilirsiniz. Bunu dalın bulunduğu sayfada "Import details" alanındaki "Import Now" düğmesi ile gerçekleştirebilirsiniz.

Paketleme işlemi için gereken tüm kodlar tek bir git projesi üzerinde olabileceği gibi birden fazla projeye de ayrılmış olabilir. Örneğin paketleme ayarlarının bulunduğu debian dizini ayrı ya da dil dosyaları ayrı bir yerde bulunuyor olabilir. Bu şekilde her bir ayrı git bağlantısı için yeni bir dal oluşturarak hepsinin senkronize olmasını sağlayın.

3.5 Projenin derlenip .deb paketi haline getirilmesi ve açtığımız ppa üzerinde yayınlanması için "recipe" oluşturma

En son kodlarımızı Launchpad üzerine çekmiştik. Geriye kalan tek şey paketin oluşturulup ppa içerisine koyulması. Bunun için kodların bulunduğu dalın sayfasını açın. (R) Orada "Related source package recipes" alanında "Create packaging recipe" bağlantısına tıklayın. Açılan sayfadaki ayarları kendinize göre değiştirebilirsiniz ama bilmiyorsanız dokunmayın. "Default distribution series:" kısmında hangi dağıtımlar için paketleme yapılacağını seçin. Unutmayın ne kadar çok seçerseniz o kadar çok beklemek zorunda kalırsınız. Son olarak "Recipe text:" kısmına aşağıdaki satırları yazın.

```
1 # bzr-builder format 0.3 deb-version {debupstream}-0~{revno}
2 lp:bootableusb
```

Eğer birden fazla dalınız var ve paketlenmesi için bu dalların birleşmesi gerekiyorsa, bu satırların altına

```
1 merge packaging lp:diğerDal
```

şeklinde diğer dallarınız için ekleme yapın. "Create recipe" düğmesi ile kaydettiğinizde tüm işlem tamamlanmış olacak ve kodlarınız paketlenmesi için sıraya alınacaktır. Bu paketleme işlemi belirli aralıklarda kodunuz değiştiğinde otomatik yapılacaktır. (S) (R) Eğer beklemek istemiyorsanız "Recipe" sayfasında "Build now" düğmesini kullanarak elinizle paketleme işlemini başlatabilirsiniz.

(S) Paketleme işlemi bittiğinde bu paketler ppa'nız içerisine aktarılacaktır.

4 Arch Linux kullanıcı deposu AUR için PKGBUILD dosyasının hazırlanması

Yazdığınız yazılımın aur deposunda listelenebilmesi için PKGBUILD dosyasına ihtiyaç vardır. Bu bölümde bu dosyanın hazırlanması gösterilecektir. Anlatım olabildiğince basit ve temel düzeydedir. Daha ayrıntılı bilgiye https://wiki.archlinux.org/index.php/PKGBUILD sayfasından ulaşabilirsiniz.

Örnek bir PKGBUILD dosyası alıp üzerinde oynama yapalım.

```
# Maintainer: bugra9 <***@gmail.com>

pkgname=bootableusb
pkgver=0.3
pkgrel=1
pkgdesc="Create bootable usb drives"
arch=('i686' 'x86_64')
url="https://github.com/bugra9/BootableUSB"
license=('GPL3')
groups=()
depends=()
makedepends=()
optdepends=()
source=("bootableusb::git+https://github.com/bugra9/BootableUSB.git")
md5sums=('SKIP')

build() {
        cd bootableusb
        make
}

package() {
        cd bootableusb
        make DESTDIR="$pkgdir" install
}
```

- **pkgname:** Paket adı
- **pkgver:** Paket sürümü
- **pkgrel:** Paket sürümünün değişmediği ama kodlarda değişiklik olduğu zaman burası bir arttırılarak sistemin paket üzerinde güncelleme olduğunun bilinmesi sağlanır. "daily build" şeklinde isimlendirilen en güncel halinin tutulduğu paketlerde kritik öneme sahiptir.
- **pkgdesc:** Paket hakkında açıklama
- **depends:** Yazılımın çalışması sırasında ihtiyaç duyacağı tüm paketler buraya yazılarak paketle beraber yüklenilmesi sağlanır. Örn: depends =('foobar>=1.8.0' 'foobar<2.0.0')
- **makedepends:** Yazılımın derlenmesi sırasında gerekli olan paketler burada belirtilir.
- **source:** Yazılımın kaynak dosyasının bulunduğu yer belirtilir. Github üzerinden çekeceğimiz için örnekte verildiği gibi kullanabilirsiniz.
- **build():** Yazılımın derlenmesi burada yazılı komutlara göre yapılır.
- **package():** Yazılımın yüklenmesi burada yazılı komutlara göre yapılır.

5 Arch Linux kullanıcı deposu AUR üzerinden yazılım yayımlanması

AUR deposuna bir yazılım ekleyebilmemiz için öncelikle ssh anahtarı oluşturup profil sayfamızda “SSH Public Key:” kısmına bu anahtarın herkese açık kısmını buraya girmeliyiz. Bundan sonra bu anahtar ile depoya erişebileceğiz. Eğer ssh anahtarını nasıl oluşturacağınızı bilmiyorsanız “seahorse” paketini yükleyip “Parolalar ve Anahtarlar” yazılımı ile görsel olarak bu anahtarı oluşturabilirsiniz.

5.1 Yeni paket oluşturma

```
1 git clone ssh://aur@aur.archlinux.org/paketAdı.git
```

komutundaki paketAdı kısmını kendinize göre değiştirip çalıştırdığınızda paketiniz için bir git deposu oluşacak ve bu depo komutu çalıştırdığınız konumda paketAdı dizinine indirilecektir.

5.2 Paketi güncelleme

Bu dizinin içerisine girip PKGBUILD dosyasını buraya kopyalayın. Daha sonra aşağıdaki komutlarla bu dosyayı uzak depomuza gönderelim.

```
1 makepkg --printsrcinfo > .SRCINFO
2 git add PKGBUILD .SRCINFO
3 git commit -m "ilk paketimi oluşturdum"
4 git push
```

Hepsi bu kadar. Şimdi https://aur.archlinux.org/packages/ bağlantısından paketinizi arayıp görüntüleyebilirsiniz. Eğer paketi güncellemek isterseniz yine aynı komutlarla bu değişikliği aktarabilirsiniz.

5.3 Paketin yüklenmesi

```
1 yaourt -S paketAdi
```

komutuyla bu oluşturduğunuz paketi bilgisayarınıza yükleyebilirsiniz.