$ cd /workspace → bu dizinde ol (workspace → Gitpod içindir. Yerelde çalışıyorsanız, workspace → yerelinizde “core”u indirmek istediğiniz herhangi bir dizindir.)

https://www.libreoffice.org/about-us/source-code/ ‘dan
$ git clone https://git.libreoffice.org/core → kaynağı indir.

$ cd → home dizininde ol
$ ssh-keygen → varsayılan yere kaydet, basit parola gir (commit yollarken bu parolayı soracak)
$ cat .ssh/id_rsa.pub → açık anahtarı ekrana yaz ve kopyala(id_rsa → gizli anahtardır.)

https://gerrit.libreoffice.org/settings/#SSHKeys → ssh key kısmına ekle.

$ cd /workspace
$ vim autogen.input
https://gist.github.com/mrkara/18cefc8b48e403df5677b03c45145dcc :
autogen.input*:*

--without-java
--without-help
--with-parallelism=8

ekle ve kaydet.

$ cp autogen.input core/ → dosyayı /core ‘a kopyala
$ cd core

https://www.libreoffice.org/about-us/source-code/ ‘dan
$ ./autogen.sh → sistemdeki gereksinimleri kontrol eder ve asıl derleme işini yapacak makefile denilen derleyici dosyalarını oluşturuyor.

Warning verirse nasm ile ilgili:
$ brew install nasm
$ ./autogen.sh → başarılı bi şekilde bitmeli. (warning olmadan)
$ make → (Çok uzun süren (2 - 4 saat) derleme işlemi başlar.)

(Yeni terminalden devam et)
$ cd /workspace/core

Gerrit ile ilgili yapılandırma ayarları:
$ ./logerrit setup → username soracak, gerrit.libreoffice.org ‘taki username ‘ini yaz.
Host ile başlayan kısmı kopyala ve
$ vim ~/.ssh/config → yapıştır, kaydet. (en son kısma newline ekle)
$ ./logerrit test → ayarlarımızı test etmek için. Ilk kez ssh sunucusu ile iletişim kuracağı için parmak izini doğrulamamızı istiyor. ‘yes’ yaz. Daha önce oluşturduğumuz ssh parolasını gir. Bi 5-10 sn bekle ve enter ‘a bas. Setup successful demeli.

Git ayarları:
$ git config --global -l → burada user.name ve user.email kontrol et.
$ git config --global user.name “Bayram Çiçek” (ya da direkt → $ vim ~/.gitconfig)
$ git config --global user.email “mail@bayramcicek.com.tr” → gerrit profildeki yazan mail ile aynı olmalı.
$ git config --global -l → doğruluğundan emin ol.

Yeni branch oluştur. (master ‘ı temiz tutmak için)
$ git checkout -b commit-second → her yeni çalışmada yeni branch aç.

Değişiklikleri yap.
$ git diff
$ git status

$ make && make check

commit mesajını editorde yazmak için bunu kullan.
$ git add ./ && $ git commit → commit mesajını editorde yaz.

commit mesajını direkt terminalde yazmak için bunu kullan.
$ git commit -a -m “tdf#88205: Adapt uses of css::uno::Sequence to use initializer_list ctor”
(The maximum length for this line is 72 characters.)

$ git status

$ git log → en üstte kendi adımızı ve değişikliğimizi görmemiz gerek.
$ ./logerrit submit master → ssh key parolasını gir. SUCCESS yazmalı ve değişikliğin linkini vermeli.

Yeni değişiklik için:
$ git checkout master
$ git status
$ git checkout -b yeni-branch
(**Değişiklikleri yap.**)

Daha önce gönderilen commit üzerinde değişiklik yapmak için:
$ git checkout ilgili-branch-adi
Değişiklikleri yap.

$ make && make check
$ git commit -a --amend → commit işlenir. (editorde açılır)
$ ./logerrit submit master

(commit yollamakta sıkıntı çıkarsa)
Gerrit üzerindeki commit’i yerele çekip düzenleme || yereldeki commiti gerrite yolla:
$ git log -1 (commit yerelde ise – SHA/commit ID kopyala)
$ git checkout -b newbranch origin/master
$ git cherry-pick <hash if the commit from git log -1> (veya gerrit üzerindeki SHA ID)
$ (düzenlemeleri yap)
$ git commit -a --amend → commit işlenir. (editorde açılır)
$ ./logerrit submit master

master branch ‘ı güncellemek için:
$ git checkout master
$ git pull -r
$ git log → son commit ‘in yakın tarihte olduğunu gör.

Start the program by running: $ instdir/program/soffice

Qt Creator Entegrasyonu:
$ cd core
$ make qtcreator-ide-integration ("Successfully created the project files." yazısını gör.)
$ Qt creator aç → "Open Project" → “core” dizini içinde → "lo.pro" dosyasını aç. Proje otomatik açılacaktır.

- Projeyi test etmek için: Build → Run

Notlar:

$ git grep -10 "…" → 10 satır aşağı ve yukarı gözükecek şekilde yazdırır.
$ git grep -n “…” → kaçıncı satırda olduğunu gösterir.
$ git grep -n isModified -- ./desktop/source/deployment/*
$ git grep -n highlight -- './sw/*' ':!./sw/qa/*' → ./qa/* hariç
$ git grep -n -10 'image' -- './*' ':!./*/qa/*' | grep 'flip'
$ find ./ -iname bulletsandnumbering.ui → dosya ismi ara

$ git stash → değişiklikleri kaydedip başka branch’a geç.
$ git stash pop → kaydedilen değişkenleri tekrar etkin et.
$ git add ./ && $ git commit → commit’i editorde yaz.
$ git commit -a --amend → en son commit’i editorde düzenle.

***

Orijinal metin : https://etherpad.gnome.org/p/ACMGitHub (Muhammet Kara)
Düzenleyen : Bayram Çiçek