**ONLYOFFICE Document Server在局域网Windows上部署和开发**

——实现企业实时文档协作

[0、说明 2](#_Toc508716172)

[1、安装准备 3](#_Toc508716173)

[2、拉取ONLYOFFICE Document Server 的Docker镜像 5](#_Toc508716174)

[3、启动——映射至9000端口 5](#_Toc508716175)

[4、替换容器onlyoffice document server的字体 6](#_Toc508716176)

[5、二次开发EngineerCMS 9](#_Toc508716177)

[5.1 协作编辑页面——汉化界面 9](#_Toc508716178)

[5.2 协作页面的保存和回调 10](#_Toc508716179)

[6、端口转发 12](#_Toc508716180)

[7、其他Linux命令解释 15](#_Toc508716181)

[7.1 主机中移除镜像 16](#_Toc508716182)

[7.2 使用“docker exec -it”命令进入容器 20](#_Toc508716183)

[7.3 退出container 21](#_Toc508716184)

[7.4 检查运行中的镜像 23](#_Toc508716185)

[7.5 下载离线镜像后导入镜像 23](#_Toc508716186)

[7.6 存出和载入镜像 23](#_Toc508716187)

[7.7 docker 自动下载的镜像放到哪了 24](#_Toc508716188)

[7.8 加速下载 24](#_Toc508716189)

[7.9 docker 镜像/容器导入导出有两种方法： 25](#_Toc508716190)

[7.10 docker load与docker import 25](#_Toc508716191)

[7.11 使用 Dockerfile 构建镜像 27](#_Toc508716192)

## 0、说明

ONLYOFFICE Document Server提供文档协作的服务功能，支持Word，Excel和PowerPoint的协作。但是[这里](https://api.onlyoffice.com/editors/howitworks)告诉我们，需要进行文档管理和存储的二次开发。Nextcloud，可道云等云盘系统也支持ONLYOFFICE协作服务，我用Golang语言开发的EngineerCMS来实现ONLYOFFICE的文档协作。

Please note, that ONLYOFFICE Document Server includes the **document editor**, **document editing service**, **document command service**and **document conversion service**. The **document manager** and **document storage service** are either included to Community Server or must be implemented by the software integrators who use ONLYOFFICE Document Server on their own server.

协作的好处在于少了来回传递文档，少了汇总的负责人，特别是十多人合作编写文档的时候，汇总的工作量大，容易出错，任何人稍微修改一点，都要重新传递整个文档，汇总后还要再次发给大家核对……。

协作也可以应用于校审流程，校核和审查直接在文档上修改即可。

其他如月报的合作，工作任务表的更新：每个人的任务表要随时更新，除非用专业的任务管理软件，否则用Excel表格的话，也是要负责人汇总更新表格，应付领导检查。

会议纪要写好了，把链接发给大家，有意见直接在上面修改补充。

投标要多人协作的话，用这个平台也是不错的。

[市场上](http://next.36kr.com/posts/collections/21)协作的平台不下几十种，著名的如石墨，一起写，Google Doc，office365，office online等，基本按人月收费，免费的有人数限制，并且想自己部署在局域网的话也不大可能。

所以，我就在ONLYOFFICE Document Server基础上用Golang语言开发了文档管理和存储功能，方便大家免费无限制任意部署的需求。

总的思路就是比如在局域网的某台windows电脑上安装docker容器来运行ONLYOFFICE Document Server，然后运行EngineerCMS来调用ONLYOFFICE Document Server的服务。在云服务器上部署也是一个道理。

ONLYOFFICE目前存在的缺点：汉字输入不是很方便，就是当输入符号的时候，它会重复将前面的汉字又输进去一次：解决办法，在设置里勾选象形文字；Excel表格里的文字格式没有删除线、上下标之类，文本框中的文字有这些格式，Word中也有；QQ拼音输入法按空格不能输出汉字，搜狗输入法可以；字体/样式选择里显示的是英文，不是汉字；打开文档默认文字检查都是打开的。但瑕不掩瑜，它的功能还是很强大的，而且随着版本的更替，会解决这些问题的。

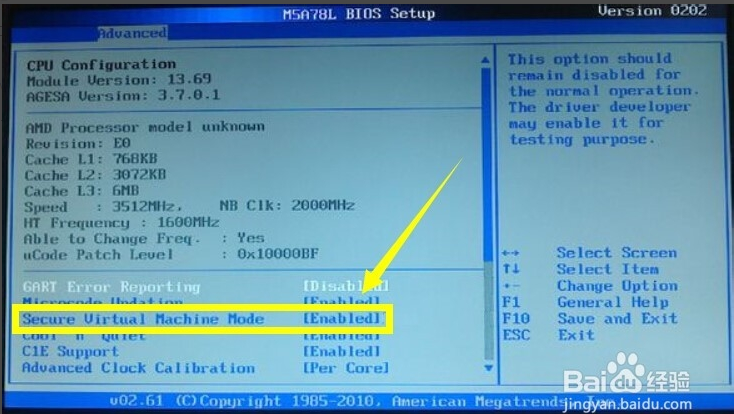
## 1、安装准备

如果是局域网，部署的电脑拥有固定ip（下文叫宿主机），可选择任何一台电脑，最好是闲置的、独立的。电脑能打开CPU的虚拟功能。如果部署到云主机，道理是一样的。

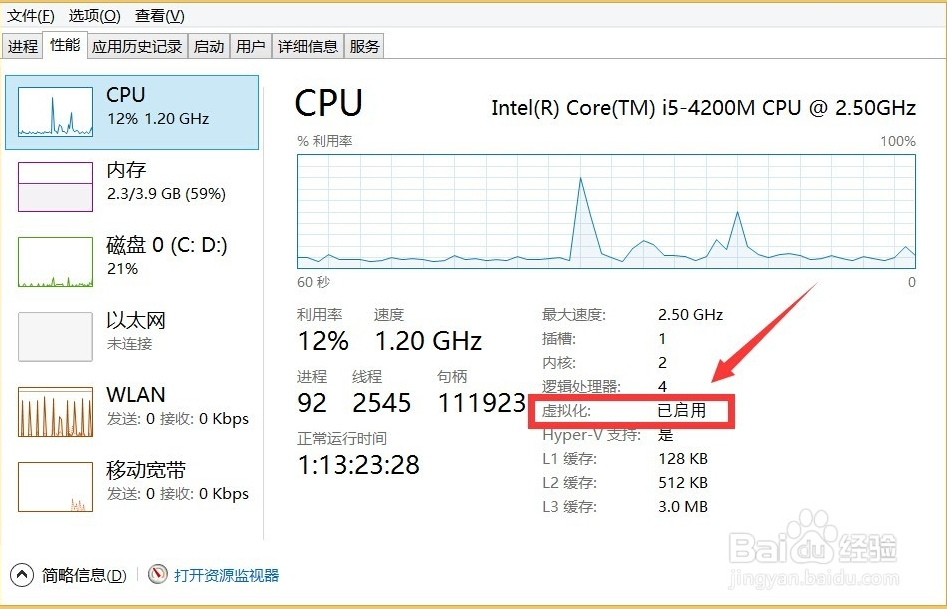
环境是win7以上，win10以下。Win10区别就是安装的docker软件不同而已。

第一步：确定电脑打开了CPU虚拟打开。<https://jingyan.baidu.com/article/22fe7ced3b4c003002617f17.html>

在CMOS界面中，切换到“Advanced”（基本设置）选项卡，找到类似“Secure VirtualMachine Mode”的项，将其设置为“Enabled”（开启）值即可。最后保存并退出CMOS界面。



从打开的“任务管理器”界面中，切换到“性能”选项卡，点击“CPU”选项，从中就可以查看到“CPU虚拟化”功能已正常开启。



第二步：下载和安装docker toolbox，默认安装即可。下载地址<http://get.daocloud.io/>，下载支持旧版windows的docker toolbox，win10就直接下载docker了。

第三步：启动docker后在里面拉取Onlyoffice Document Server 的Docker镜像

## 2、拉取ONLYOFFICE Document Server 的Docker镜像

拉取镜像：启动docker，在里面输入命令 docker pull onlyoffice/documentserver

可以用我做好的镜像导入。见docker load < documentserver.tar命令。

~~建议用离线的，已经设置好了字体，可以省略第4步，直接到第5步：~~

~~（Administrator@604TFALNDKDKJWC MINGW64 ~~~

~~$ docker export -o onlyoffice-`date +%Y%m%d`.tar 6df）~~

~~$ docker import onlyoffice-20180220-import.tar onlyoffice/documentserver~~

离线包有2个，onlyoffice-20180220-import.tar这个包虽然里面已经替换好了字体，但导入后暂时不知道怎样运行起来。

onlyofficedocumentserver-load.tar这个包就是原生态的pull下来的镜像打的包，用save命令打的包，用loader命令导入，然后运行，然后替换字体。

离线的包在我的网盘中：<https://pan.baidu.com/s/1gf0ucuR>

## 3、启动——映射至9000端口

docker run -i -t -d -p 9000:80 onlyoffice/documentserver

~~PORTS 443/tcp, 0.0.0.0:9000->80/tcp~~

~~如果是用import导入的，请使用下面命令：~~

docker run -i -t -d -p 9000:80 onlyoffice/documentserver /bin/sh -c 'bash -C'

~~原理是~~[~~https://www.cnblogs.com/wish123/p/6573899.html~~](https://www.cnblogs.com/wish123/p/6573899.html)~~：~~

~~docker run [OPTIONS] IMAGE [COMMAND] [ARG...]~~



## 4、替换容器onlyoffice document server的字体

在网盘中下载字体winfont压缩包。字体来源于<https://github.com/neroxps/Docker-Only-Office-Chinese-font>和win10系统，win10系统提取中文字体的方法：控制面板——搜字体——查看安装的字体——再在搜索栏输入 中文 2个字，这些就是需要的中文字体了。

//进入容器（运行的镜像）内，删除/usr/share/fonts 下除truetype外其他文件和文件夹

**$** **docker exec -it 38e27 /bin/bash**

root@38e27823ae92:/# dir或者ls –al

root@6dfa5705aaef:~# cd /usr/share/fonts/

root@6dfa5705aaef:/usr/share/fonts# ls

truetype X11

//删除文件夹X11

root@6dfa5705aaef:/usr/share/fonts# **rm -R dir X11**

rm: cannot remove dir: No such file or directory

root@6dfa5705aaef:/usr/share/fonts# ls

truetype

root@6dfa5705aaef:/usr/share/fonts# **cd truetype**

root@6dfa5705aaef:/usr/share/fonts/truetype# ls -al

total 462392

drwxr-xr-x 11 root root 4096 Feb 19 04:17 .

………………

//删除trutype文件夹下所有文件，除了custome文件夹外

root@6dfa5705aaef:/usr/share/fonts/truetype# **rm -R dir \*.\***

rm: cannot remove dir: No such file or directory

root@6dfa5705aaef:/usr/share/fonts/truetype# **rm -R dir \***

rm: cannot remove dir: No such file or directory

rm: cannot remove custom: Device or resource busy

root@6dfa5705aaef:/usr/share/fonts/truetype# ls

custom

root@6dfa5705aaef:/usr/share/fonts/truetype# ls -al

total 12

drwxr-xr-x 10 root root 4096 Feb 19 10:14 .

drwxr-xr-x 6 root root 4096 Feb 19 10:12 ..

drwxr-xr-x 2 root root 4096 Feb 19 03:48 custom

……

root@6dfa5705aaef:/usr/share/fonts/truetype# **exit**

exit

Administrator@604TFALNDKDKJWC MINGW64 ~/winfont

//将当前文件夹C:\Users\Administrator\下的winfont文件夹内的字体全部拷贝到容器的文件夹/usr/share/fonts/truetype中

字体文件在我的网盘中：<https://pan.baidu.com/s/1gf0ucuR>

$ **tar -cv \* | docker exec -i 6df tar x -C /usr/share/fonts/truetype**

kaiu.ttf

msjh.ttc

msjhbd.ttc

msjhl.ttc

msyh.ttc

msyh.ttf

msyhbd.ttc

msyhl.ttc

simfang.ttf

simhei.ttf

simkai.ttf

simli.ttf

simsun.ttc

simsunb.ttf

simyou.ttf

……

Administrator@604TFALNDKDKJWC MINGW64 ~/winfont

//进入容器内

$ **docker exec -it 6df /bin/bash**

root@6dfa5705aaef:/# **sudo mkfontscale**

root@6dfa5705aaef:/# **sudo mkfontdir**

root@6dfa5705aaef:/# **sudo fc-cache -fv**

/usr/share/fonts: caching, new cache contents: 0 fonts, 1 dirs

…………

fc-cache: succeeded

root@6dfa5705aaef:/# **exit**

exit

//退出容器

Administrator@604TFALNDKDKJWC MINGW64 ~/winfont

$ **docker exec 6df /usr/bin/documentserver-generate-allfonts.sh**

Generating AllFonts.js, please wait...Done

onlyoffice-documentserver:docservice: stopped

onlyoffice-documentserver:docservice: started

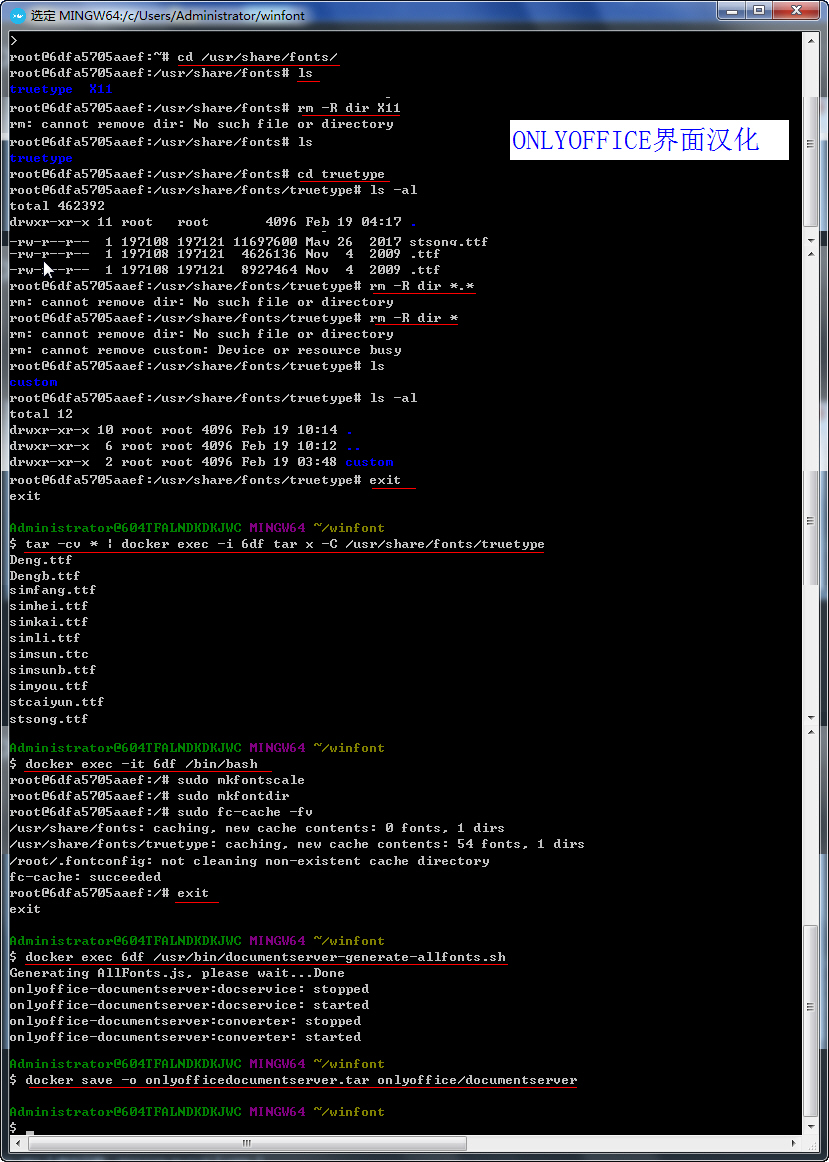
onlyoffice-documentserver:converter: stopped

onlyoffice-documentserver:converter: started

如果机器重启后，不要用docker run命令，要用docker start 容器id

具体替换字体的命令见下图，很详细具体了。

完成后记得清理chrome浏览器缓存，否则还是乱码。设置—高级—隐私设置和安全性—清除浏览数据—时间范围选择时间不限……



## 5、二次开发EngineerCMS

直接去<https://github.com/3xxx/EngineerCMS>查看源码

直接去<https://github.com/3xxx/EngineerCMS/releases>下载编译好的二进制文件

或者直接去<https://pan.baidu.com/s/1gf0ucuR>下载软件包，运行engineercms即可用浏览器chrome访问了。

（1） 运行前要替换EngineerCMS\views\onlyoffice\onlyoffice.tpl中的3处IP，如果engineercms和docker安装在同一个电脑上，那么三个IP相同，见5.1节。

**（2）访问前要按照第6节中的方法设置宿主机（安装docker的电脑）的端口转发。**

有问题联系我QQ504284，微信hotqin999

<http://blog.csdn.net/hotqin888>

下文的开发仅供兴趣参考。

### 5.1 协作编辑页面——汉化界面

EngineerCMS\views\onlyoffice\onlyoffice.tpl：

<!DOCTYPE html>

<html style="height: 100%;">

<head>

<title>fei-OnlyOffice</title>

</head>

<body style="height: 100%; margin: 0;">

<div id="placeholder" style="height: 100%"></div>

<script type="text/javascript" src="http://**安装docker电脑的ip**:9000/web-apps/apps/api/documents/api.js"></script>

<script type="text/javascript">

// alert({{.Doc.FileName}});

window.docEditor = new DocsAPI.DocEditor("placeholder",

{

"document": {

"fileType": "{{.fileType}}",

"key": "{{.Key}}",//"Khirz6zTPdfd7"

"title": "{{.Doc.FileName}}",

"url": "http://**运行engineercms电脑的ip**/attachment/onlyoffice/{{.Doc.FileName}}"

},

"documentType": "{{.documentType}}",

"editorConfig": {

"callbackUrl": "http:// **运行engineercms电脑的ip** /url-to-callback?id={{.Doc.Id}}",

"user": {

"id": "{{.Uid}}",

"name": "{{.Uname}}"

},

"lang": "zh-CN",//"en-US",汉化

},

"height": "100%",

"width": "100%"

});

</script>

</body>

</html>

### 5.2 协作页面的保存和回调

type Callback struct {

Key string `json:"key"`

Status int `json:"status"`

Url string `json:"url"`

Changesurl string `json:"changesurl"`

History history `json:"history"`

Users []string `json:"users"`

Actions []action `json:"actions"`

Lastsave string `json:"lastsave"`

Notmodified bool `json:"notmodified"`

}

type action struct {

Type int

userid string

}

type history struct {

changes []change

serverVersion string

}

type change struct {

created string

User User1

}

type User1 struct {

id string

name string

}

//关闭浏览器标签后获取最新文档保存到文件夹

func (c \*OnlyController) UrltoCallback() {

// pk1 := c.Ctx.Input.RequestBody

id := c.Input().Get("id")

//pid转成64为

idNum, err := strconv.ParseInt(id, 10, 64)

if err != nil {

beego.Error(err)

}

//根据附件id取得附件的prodid，路径

onlyattachment, err := models.GetOnlyAttachbyId(idNum)

if err != nil {

beego.Error(err)

}

var callback Callback

json.Unmarshal(c.Ctx.Input.RequestBody, &callback)

if callback.Status == 1 || callback.Status == 4 {

c.Data["json"] = map[string]interface{}{"error": 0}

c.ServeJSON()

} else if callback.Status == 2 {

resp, err := http.Get(callback.Url)

if err != nil {

beego.Error(err)

}

body, err := ioutil.ReadAll(resp.Body)

if err != nil {

beego.Error(err)

}

defer resp.Body.Close()

if err != nil {

beego.Error(err)

}

f, err := os.OpenFile(".\\attachment\\onlyoffice\\"+onlyattachment.FileName, os.O\_RDWR|os.O\_CREATE|os.O\_APPEND, os.ModePerm)

if err != nil {

beego.Error(err)

}

defer f.Close()

\_, err = f.Write(body)

if err != nil {

beego.Error(err)

} else {

err = models.UpdateOnlyAttachment(idNum)

if err != nil {

beego.Error(err)

}

}

c.Data["json"] = map[string]interface{}{"error": 0}

c.ServeJSON()

}

}

\*\*\*\*\*\*\*\*结束\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## 6、端口转发

<http://blog.csdn.net/maodou95838/article/details/78194830?locationNum=1&fps=1#0-qzone-1-51693-d020d2d2a4e8d1a374a433f596ad1440>

前文第3节运行ONLYOFFICE DOCUMENT SERVER：

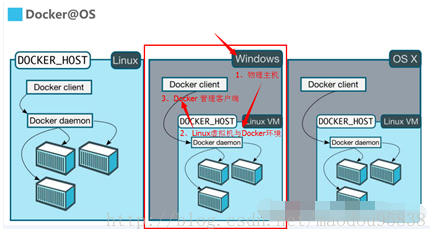
docker run -i -t -d -p 9000:80 onlyoffice/documentserver

**-d:后台运行**

**-p 9000:80 :端口映射，前一个9000代表虚拟机的端口，后一个80代表docker容器的端口**

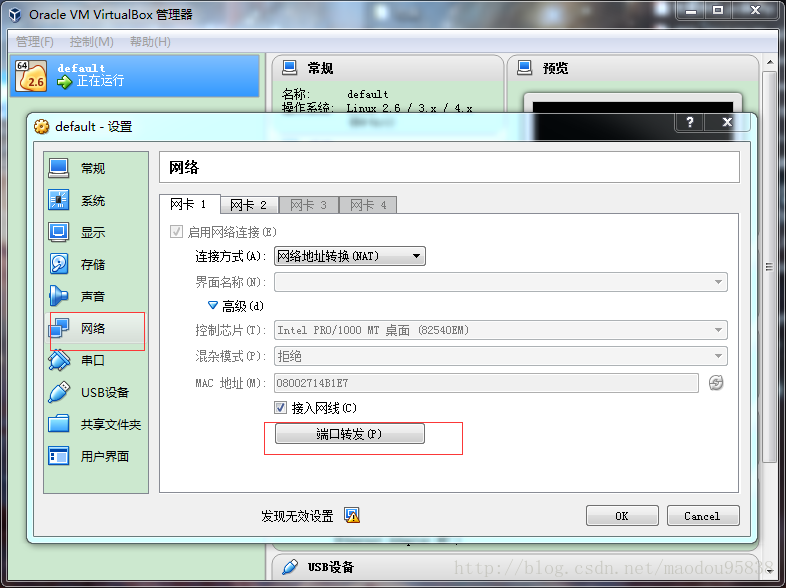
命令执行完成后，可以通过<http://192.168.99.100:9000>访问。

这时只能在你自己的机器上访问，同一个局域网中的其他机器是不能访问的，原因是：Windows上的docker是运行在一个虚拟机里面的，看官方给的图：



要让别的机器也访问到，需要配置一下虚拟机。设置如下：

双击运行安装docker toolbox生成的Oracle VM VirtualBox。



点击端口转发：**下图中主机IP为安装docker的电脑IP，主机端口为9000，子系统IP为192.168.99.100，子系统端口为9000**。





这样配置完成后，别的机器也可以访问docker 容器了。

## 修正excel翻译错误：

Administrator@604TFALNDKDKJWC MINGW64 ~/zh

$ docker exec -it 6df /bin/bash

root@6dfa5705aaef:/# cd /var/www/onlyoffice/documentserver/web-apps/apps/spreadsheeteditor/main/locale/

root@6dfa5705aaef:/var/www/onlyoffice/documentserver/web-apps/apps/spreadsheeteditor/main/locale# ls

cs.json en.json fr.json ja.json nl.json pt.json sk.json tr.json vi.json

de.json es.json it.json lv.json pl.json ru.json sl.json uk.json zh.json

root@6dfa5705aaef:/var/www/onlyoffice/documentserver/web-apps/apps/spreadsheeteditor/main/locale# rm zh.json

root@6dfa5705aaef:/var/www/onlyoffice/documentserver/web-apps/apps/spreadsheeteditor/main/locale# exit

exit

Administrator@604TFALNDKDKJWC MINGW64 ~/zh

$ tar -cv \* | docker exec -i 6df tar x -C /var/www/onlyoffice/documentserver/web-apps/apps/spreadsheeteditor/main/

locale/

zh.json



word的翻译文件/var/www/onlyoffice/documentserver/web-apps/apps/documenteditor/main/locale/

## 7、其他Linux命令解释

find / -name arial.ttf找到字体所在目录

root@6dfa5705aaef:/usr/share/fonts/truetype/msttcorefonts#

删除dirname目录下的所有东西

[root]# rm -R dir dirname删除所有

[root]# rm -f \*.ttf

查看镜像

列出本地主机上的镜像。你可以使用 docker images 来完成

我们在镜像列表中看到三个至关重要的东西。

* 来自什么镜像源，例如 ubuntu
* 每个镜像都有标签(tags)，例如 14.04
* 每个镜像都有镜像ID

镜像源中可能存储着一个镜像源的多个版本。我们会看到 Ubuntu 的多个版本：10.04, 12.04, 12.10, 13.04, 13.10 and 14.04。每个容器有一个唯一的标签(tag)，让我们来识别为不同的镜像，例如：

ubuntu:14.04

所以我们可以运行一个带标签镜像的容器：

$ sudo docker run -t -i ubuntu:14.04 /bin/bash

如果我们想要使用 Ubuntu 12.04 的镜像来构建，我们可以这样做

$ sudo docker run -t -i ubuntu:12.04 /bin/bash

如果你不指定一个镜像的版本标签，例如你只使用 Ubuntu，Docker 将默认使用 Ubuntu:latest 镜像。

### 7.1 主机中移除镜像

你可以删除你主机上的镜像，类似于删处容器的方法，这里我们使用 docker rmi 命令。

让我们删除这个不需要使用的容器：training/sinatra。

$ docker rmi training/sinatra

#### 7.1.1 删除Docker镜像

首先要保证有权限对Docker镜像或者容器进行操作，具体做法参见进入前文安装部分。

# 停止所有容器

$ docker stop $(docker ps -a -q)

# 删除指定镜像

$ docker rmi $image

# 删除无标示镜像，即id为<None>的镜像

$ docker rmi $(docker images | grep "^<none>" | awk "{print $3}")

# 删除所有镜像

$ docker rmi $(docker images -q)

查看容器

#### 7.1.2 列出容器

$ sudo docker ps # Lists only running containers

$ sudo docker ps -a # Lists all containers

$ sudo docker ps -l # List the last running container

启动容器

#### 7.1.3 控制容器

# 开始一个新的容器

$ JOB=$(sudo docker run -d ubuntu /bin/sh -c "while true; do echo Hello world; sleep 1; done")

# 停止容器

$ docker stop $JOB

# 开始容器

$ docker start $JOB

# 重启容器

$ docker restart $JOB

# 杀死一个工作

$ docker kill $JOB

# 删除一个容器

$ docker stop $JOB # Container must be stopped to remove it

$ docker rm $JOB

* 查看本机images

docker images

* 查看本机运行的container

docker ps

* 从dockerhub上pull 拉取镜像

docker pull 镜像名称

* 利用image运行container：

docker run -it --rm -p 8888:8888 -v /c/Users/tingting/dropbox/code:/root/opt/workspace -v /c/Users/tingting/dropbox/data:/root/data tingtinglu/caffe\_mxnet

（1）利用参数 -v

（2）冒号前为宿主机目录，必须为绝对路径

① /c/Users/tingting/dropbox/code:

本机的C:\Users\tingting\dropbox\code文件夹

② /c/Users/tingting/dropbox/data

本机的C:\Users\tingting\dropbox\data文件夹

（3）冒号后为镜像内挂载的路径

① /root/opt/workspace

docker中的文件夹/root/opt/workspace

② /root/data

docker中的文件夹/root/data

（4）这里的本机文件夹为c盘，那么，能否为d盘呢？

测试：

docker run -it --rm -p 8888:8888 -v /d/Dropbox/code:/root /opt/workspace -v /d/Dropbox/data:/root/data tingtinglu/caffe\_mxnet

发现并不可行！这是因为目前，windows下只支持c盘下的文件夹映射

退出容器到docker中来：exit

Administrator@604TFALNDKDKJWC MINGW64 ~/winfont

$ tar -cv \* | docker exec -i 38e tar x -C /usr/share/fonts/truetype/msttcorefonts

msyh.ttf

wqy-microhei.ttc

wqy-zenhei.ttc

wqy-zenhei.ttf

拷贝字体过去

再进入容器

docker exec -it 38e27 /bin/bash

sudo mkfontscale （创建雅黑字体的fonts.scale文件，它用来控制字体旋转缩放）

sudo mkfontdir （创建雅黑字体的fonts.dir文件，它用来控制字体粗斜体产生）

sudo fc-cache -fv （建立字体缓存信息，也就是让系统认识雅黑）

退出到docker：exit

$ docker exec 38e27 /usr/bin/documentserver-generate-allfonts.sh

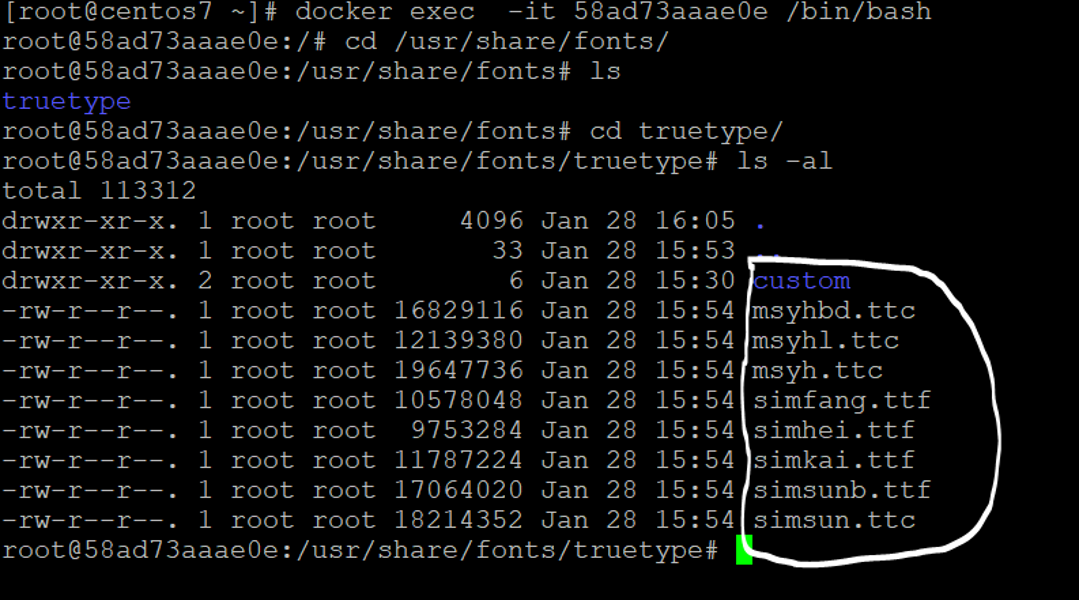
Generating AllFonts.js, please wait...Done

onlyoffice-documentserver:docservice: stopped

onlyoffice-documentserver:docservice: started

onlyoffice-documentserver:converter: stopped

onlyoffice-documentserver:converter: started

[**@JoeLucky**](https://github.com/joelucky) yes，i copy some Chinese fonts into your system，and delete all others，clear browser cache，at the end everything will be work perfect。  
[](https://user-images.githubusercontent.com/17507066/35509246-6ce2555c-052e-11e8-8c8f-cd144b8b526e.png)

root@38e27823ae92:/usr/share/fonts/truetype/msttcorefonts#

docker run -it -v /home/dock/Downloads:/usr/Downloads ubuntu64 /bin/bash

38e27823ae92

docker run -t -i ubuntu /bin/bash

[官网](https://docs.docker.com/engine/tutorials/dockerizing/)是这么说的：

* **docker run: runs a container.**
* **ubuntu: is the image you would like to run.**
* **-t: flag assigns a pseudo-tty or terminal inside the new container.**
* **-i: flag allows you to make an interactive connection by grabbing the standard in (STDIN) of the container.**
* **/bin/bash: launches a Bash shell inside our container.**

理解很简单：

docker run：启动container

ubuntu：你想要启动的image

-t：进入终端

-i：获得一个交互式的连接，通过获取container的输入

/bin/bash：在container中启动一个bash shell

这样就进入container的内部了：root@af8bae53bdd3:/#

### 7.2 使用“docker exec -it”命令进入容器

使用“docker attach”命令进入container（容器）有一个缺点，那就是每次从container中退出到前台时，container也跟着退出了。

要想退出container时，让container仍然在后台运行着，可以使用“docker exec -it”命令。每次使用这个命令进入container，当退出container后，container仍然在后台运行，命令使用方法如下：docker exec -it goofy\_almeida /bin/bash

* **goofy\_almeida：要启动的container的名称**
* **/bin/bash：在container中启动一个bash shell**

这样输入“exit”或者按键“Ctrl + C”退出container时，这个container仍然在后台运行，通过：docker ps就可以查找到。

### 7.3 退出container

输入：exit

或者按键：Ctrl + C

3 以下就是解决办法：

   1 首先上传自己喜欢的中文字体到document server容器的/usr/share/fonts/chinese目录下（默认没有此目录，需自行创建）。

    2 执行以下命令,建立字体缓存信息：

       cd /usr/share/fonts/Chinese——根据上述命令，进入容器内部可以执行以下命令

       apt-get install xfonts-utils

        mkfontscale  
　　 mkfontdir  
        apt-get install fontconfig  
　　 fc-cache -fv  
　　 命令执行成功后终端上会显示：  
　　 fc-cache: succeeded

        然后，执行document server容器自带的脚本  
            documentserver-generate-allfonts.sh

      Then clear cache browser and logon again.

[**@bdxing**](https://github.com/bdxing) 贴一个解决办法[**@eqmo**](https://github.com/eqmo)的解决办法  
首先将.ttf字体放入fonts文件夹  
然后：  
cd /usr/share/fonts/  
tar cv \* | docker exec -i XXX tar x -C /usr/share/fonts/  
docker exec XXX /var/www/onlyoffice/documentserver/Tools/GenerateAllFonts.sh  
XXX是容器id

然后重启容器，清理浏览器缓存就好了，另外服务完全重启需要几分钟时间，如果重启不成功会出现打开文档报错，或者无法打开等问题。可以重新重启容器。

The new version the command after copy the fonts is different:  
Run this command against the document server container ID XXXX

docker exec XXXXXX /usr/bin/documentserver-generate-allfonts.sh

docker exec 9cfbb /usr/bin/documentserver-generate-allfonts.sh

docker ps

Please follow this instruction.

1. Copy the font to 'C:\Windows\Font' (to the computer with Document Server installed).
2. Open the folder 'C:\Program Files\ONLYOFFICE\DocumentServer\bin' and run 'documentserver-generate-allfonts.bat'.  
   It will scan the system font folder for the installed font. The font will be used by the AllFonts.js script to form the font thumbnails in the editors and add the font metrics to the font\_selection.bin file for its further usage.
3. Find in Task Manager 'ONLYOFFICE Converter Service' and restart it.
4. Clear the browser cache and reopen the page.

登入容器，也可以将以上两个目录映射到本地，这样更方便。

创建目录：

mkdir -p /app/onlyoffice/DocumentServer/logs

mkdir -p /app/onlyoffice/DocumentServer/data

C:\Program Files\Git\app\onlyoffice

使用

docker -v

 将目录映射至本地：

映射目录至本地

docker run -i -t -d -p 80:80 \

    -v /app/onlyoffice/DocumentServer/logs:/var/log/onlyoffice  \

    -v /app/onlyoffice/DocumentServer/data:/var/www/onlyoffice/Data  onlyoffice/documentserver

### 7.4 检查运行中的镜像

现在你已经运行了一个docker容器，让我们来看下正在运行的容器。

使用docker ps命令可以查看所有正在运行中的容器列表，使用docker inspect命令我们可以查看更详细的关于某一个容器的信息。

**目标：**

查找某一个运行中容器的id，然后使用docker inspect命令查看容器的信息。

**提示：**

可以使用镜像id的前面部分，不需要完整的id。

正确的命令：

$ **docker inspect efe**

### 7.5 下载离线镜像后导入镜像

cat fee97b71-17d7-4fab-a5b0-69d4112521e6 | docker **import** - cloudfocker-base

### 7.6 存出和载入镜像

存出镜像

如果要导出镜像到本地文件，可以使用 docker save 命令。

docker save -o ubuntu\_14.04.tar ubuntu:14.04

载入

docker load --input ubuntu\_14.04.tar

docker load < ubuntu\_14.04.tar这将导入镜像以及其相关的元数据信息（包括标签等）

从本地文件系统导入

要从本地文件系统导入一个镜像，可以使用 openvz（容器虚拟化的先锋技术）的模板来创建： openvz 的模板下载地址为 [templates](http://openvz.org/Download/templates/precreated) 。

比如，先下载了一个 ubuntu-14.04 的镜像，之后使用以下命令导入：

sudo cat ubuntu-14.04-x86\_64-minimal.tar.gz |docker import - ubuntu:14.04

然后查看新导入的镜像。

docker images

docker load 和 上一节中从本地导入 镜像有什么区别吗,我怎么感觉是差不多的呢

不一样的。看：https://stackoverflow.com/questions/36925261/what-is-the-difference-between-import-and-load-in-docker

docker import只会从压缩文件中导入一个 image

而 docker load 则会从压缩文件中导入多个 image

### 7.7 [docker 自动下载的镜像放到哪了](https://segmentfault.com/q/1010000003005948)

/var/lib/docker

实际存放images和container的地方就是/var/lib/docker/devicemapper/devicemapper目录下的metadata和data两个文件。

windows上的docker本质上还是借助与windows平台的hyper-v创建一个linux虚拟机，你执行的所有命令都是在这个虚拟机里执行的，所有pull到本地的image都会在虚拟机的Virtual hard disks目录的文件中，这个文件就是虚拟硬盘文件（有点类似与vmware的原理）。  
因此你打开hyper-v管理器，可以找到docker创建的虚拟机，点击左侧的虚拟机名称，然后再点击右边的移动选项，按照向导将虚拟机移动到其他目录即可。  
另外还可以在菜单栏点击:操作->Hyper-v设置，来调整你所有虚拟机的虚拟硬盘文件的默认存储位置

### 7.8 加速下载

# docker pull daocloud.io/library/centos:latest

### 7.9 docker 镜像/容器导入导出有两种方法：

一种是使用 save 和 load 命令——用于导出镜像

使用例子如下：镜像名称

~~docker save ubuntu:load > /root/ubuntu.tar~~

~~docker load < ubuntu.tar~~

一种是使用 export 和 import 命令——用于导出容器最终状态

使用例子如下：容器id

~~docker export 98ca36 > ubuntu.tar~~

~~cat ubuntu.tar | sudo docker import - ubuntu:import~~

### 7.10 docker load与docker import

  首先，想要清楚的了解docker load与docker import命令的区别，就必须了解镜像与容器的区别：

* 镜像：用来启动容器的只读模板，是容器启动所需的rootfs，类似于虚拟机所使用的镜像。
* 容器：Docker 容器是一个开源的应用容器引擎，让开发者可以打包他们的应用以及依赖包到一个可移植的容器中，然后发布到任何流行的Linux机器上，也可以实现虚拟化。

镜像是容器的基础，可以简单的理解为镜像是我们启动虚拟机时需要的镜像，容器时虚拟机成功启动后，运行的服务。

  想要了解docker load与docker import命令的区别，还必须知道docker save与docker export命令：

* docker save images\_name：将一个镜像导出为文件，再使用docker load命令将文件导入为一个镜像，会保存该镜像的的所有历史记录。比docker export命令导出的文件大，很好理解，因为会保存镜像的所有历史记录。
* docker export container\_id：将一个容器导出为文件，再使用docker import命令将容器导入成为一个新的镜像，但是相比docker save命令，容器文件会丢失所有元数据和历史记录，仅保存容器当时的状态，相当于虚拟机快照。

如果要导出本地某个容器，可以使用 **docker export** 命令，可以使用 **docker import** 从容器快照文件中再导入为镜像

docker export [OPTIONS] CONTAINER

OPTIONS说明：

* **-o :**将输入内容写到文件。

runoob@runoob:~$ docker export -o mysql-`date +%Y%m%d`.tar a404c6c174a2

docker import [OPTIONS] file|URL|- [REPOSITORY[:TAG]]

OPTIONS说明：

* **-c :**应用docker 指令创建镜像；
* **-m :**提交时的说明文字；

**实例**

从镜像归档文件my\_ubuntu\_v3.tar创建镜像，命名为runoob/ubuntu:v4

runoob@runoob:~$ docker import my\_ubuntu\_v3.tar runoob/ubuntu:v4

我个人粗略总结了下步骤：确保onlyoffice已经正常运行状态中。

 1、调出终端 find / -name arial.ttf

可查到onlyoffice的字库文件夹fonts，路径很长，删空里面文件之后，  
打开nextcloud之中的文档，字体已经为空。记得先删除浏览器缓存再打开文档。  
 2、可参考网站C:\Users\ADMINI~1\AppData\Local\Temp\[5UQ[BL(6~BS2JV6W}N6[%S.pnghttp://wiki.ubuntu.org.cn/

字体，调出终端，安装文泉驿-微米黑等多种中文字体（路径：/usr/share/fonts/truetype/wqy），再把字体拷贝到onlyoffice的fonts文件夹内。

步骤：譬如安装的字体路径在/usr/share/fonts/ truetype/wqy /下  
 3、所拷贝字体需创建缓存信息。

   cd /usr/share/fonts/truetype/wqy （进入字体所在文件夹）  
sudo mkfontscale （创建雅黑字体的fonts.scale文件，它用来控制字体旋转缩放）

sudo mkfontdir （创建雅黑字体的fonts.dir文件，它用来控制字体粗斜体产生）

sudo fc-cache -fv （建立字体缓存信息，也就是让系统认识雅黑）  
然后输入docker  ps  回车查询容器ID

然后把容器ID填进去后再运行命令：

  docker exec -it 容器ID C:\Users\ADMINI~1\AppData\Local\Temp\%W@GJ$ACOF(TYDYECOKVDYB.pngbashdocumentserver-generate-allfonts.sh

中文  
主条目：免费中文字体  
sudo apt-get install ttf-wqy-microhei  #文泉驿-微米黑  
sudo apt-get install ttf-wqy-zenhei  #文泉驿-正黑  
sudo apt-get install xfonts-wqy #文泉驿-点阵宋体

通过博主的教程安装了office，但是不满足于打开默认英文。于是我删除了fonts文件夹并按命令更新字体缓存行事，但是打开onlyoffice还是一样有arial字体。所以调出终端 find / -name arial.ttf ，意外发现onlyoffice字体所在文件夹，删除它们并将我下载的中文字体放进去，更新缓存等命令之后，打开完美默认中文字体。博主可以更新下onlyoffice那篇教程

### 7.11 使用 Dockerfile 构建镜像

使用 docker commit 命令能够非常简单的扩展镜像。但是它有点麻烦，并且在一个团队中也不能够轻易的共享它的开发过程。为解决这个问题，我们使用一个新的命令 docker build ， 从零开始来创建一个新的镜像。

为此，我们需要创建一个 Dockerfile 文件，其中包含一组指令来告诉 Docker 如何构建我们的镜像。

现在创建一个目录，并且创建一个 Dockerfile

$ mkdir sinatra

$ cd sinatra

$ touch Dockerfile

如果你是在 Windows 上使用的 Boot2Docker，你可以通过使用 cd 命令来访问你的本地主机目录 /c/Users/your\_user\_name

每一个指令都会在镜像上创建一个新的层，来看一个简单的例子，我们的开发团建来构建一个自己的 Sinatra镜像：

# This is a comment

FROM ubuntu:14.04

MAINTAINER Kate Smith <ksmith@example.com>

RUN apt-get update && apt-get install -y ruby ruby-dev

RUN gem install sinatra

让我们看一下我们的 Dockerfile 做了什么。每一个指令的前缀都必须是大写的。

INSTRUCTION statement

提示：我们可以使用 **#** 来注释

第一个指令 FROM 是告诉 Docker 使用的哪个镜像源，在这个案例中，我们使用的是 Ubuntu 14.04 基础镜像。

下一步，我们使用 MAINTAINER 指令来指定谁在维护这个新镜像。

最后，我们指定了两个 RUN 指令。 RUN 指令在镜像内执行一条命令，例如：安装一个包。这里我们更新了 APT 的缓存，并且安装 Ruby 和 RubyGems ，然后使用 gem 安装 Sinatra 。

注意：我们还提供了更多的 Dockerfile 指令参数。

现在，我们使用 Dockerfile 文件，通过 docker build 命令来构建一个镜像。

$ docker build -t ouruser/sinatra:v2 .

Sending build context to Docker daemon 2.048 kB

Sending build context to Docker daemon

Step 0 : FROM ubuntu:14.04

---> e54ca5efa2e9

Step 1 : MAINTAINER Kate Smith <ksmith@example.com>

---> Using cache

---> 851baf55332b

Step 2 : RUN apt-get update && apt-get install -y ruby ruby-dev

---> Running in 3a2558904e9b

Selecting previously unselected package libasan0:amd64.

(Reading database ... 11518 files and directories currently installed.)

Preparing to unpack .../libasan0\_4.8.2-19ubuntu1\_amd64.deb ...

我们使用 docker build 命令并指定 -t 标识(flag)来标示属于 ouruser ，镜像名称为 sinatra,标签是 v2。

如果 Dockerfile 在我们当前目录下，我们可以使用 . 来指定 Dockerfile

提示：你也可以指定 Dockerfile 路径

现在我们可以看到构建过程。Docker 做的第一件事是通过你的上下文进行构建，基本上是目录的内容构建。这样做是因为 Docker 进程构建镜像是实时构建的，并且是需要本地的上下文来做这些工作的。（这里上下文是指Context）

下一步，Dockerfile 中的每一条命令都一步一步的被执行。我们会看到每一步都会创建一个新的容器，在容器内部运行指令并且提交更改 - 就像我们之前使用的 docker commit 一样。当所有的指令执行完成之后，我们会得到97feabe5d2ed 镜像（也帮助标记为 ouruser/sinatra:v2）, 然后所有中间容器会被清除。

注意：与存储程序驱动无关，镜像不能超过127层。这是一种全局设置，为了是从整体上来优化镜像的大小。

我们可以使用新的镜像来创建容器：

$ docker run -t -i ouruser/sinatra:v2 /bin/bash

root@8196968dac35:/#

注意：这里只是简单的介绍一下如何创建镜像。我们跳过了很多你可以使用的其它指令。你会从后边的章节看到更多的指令或者你可以参考 Dockerfile 的详细说明和每一条指令的例子。为了帮助你编写清晰、易读、易维护的 Dockerfile ，我们也编写了 Dockerfile [最佳实践指南](http://www.widuu.com/chinese_docker/articles/dockerfile_best-practices.md)

