```
library(tm)
#Create Corpus - CHANGE PATH AS NEEDED
docs <- Corpus(DirSource("C:/Users/<YourPath>/Documents/TextMining"))
#Check details
inspect(docs)
#inspect a particular document
writeLines(as.character(docs[[30]]))
#Start preprocessing
toSpace <- content_transformer(function(x, pattern) { return (gsub(pattern, " ",
x))})
docs <- tm_map(docs, toSpace, "-")
docs <- tm_map(docs, toSpace, ":")
docs <- tm_map(docs, toSpace, "'")
docs <- tm_map(docs, toSpace, "'")
docs <- tm_map(docs, toSpace, " -")
#Good practice to check after each step.
writeLines(as.character(docs[[30]]))
#Remove punctuation - replace punctuation marks with " "
docs <- tm_map(docs, removePunctuation)
#Transform to lower case
docs <- tm_map(docs,content_transformer(tolower))
#Strip digits
docs <- tm_map(docs, removeNumbers)
#Remove stopwords from standard stopword list (How to check this? How to add your
own?)
docs <- tm_map(docs, removeWords, stopwords("english"))
#Strip whitespace (cosmetic?)
docs <- tm_map(docs, stripWhitespace)
#inspect output
writeLines(as.character(docs[[30]]))

#Need SnowballC library for stemming
library(SnowballC)
#Stem document
docs <- tm_map(docs,stemDocument)
#some clean up
docs <- tm_map(docs, content_transformer(gsub),
               pattern = "organiz", replacement = "organ")

docs <- tm_map(docs, content_transformer(gsub),
               pattern = "organis", replacement = "organ")

docs <- tm_map(docs, content_transformer(gsub),
               pattern = "andgovern", replacement = "govern")

docs <- tm_map(docs, content_transformer(gsub),
               pattern = "inenterpris", replacement = "enterpris")

docs <- tm_map(docs, content_transformer(gsub),
               pattern = "team-", replacement = "team")

#inspect
writeLines(as.character(docs[[30]]))

#Create document-term matrix
dtm <- DocumentTermMatrix(docs)
#inspect segment of document term matrix
inspect(dtm[1:2,1000:1005])
#collapse matrix by summing over columns - this gets total counts (over all docs)
for each term
freq <- colSums(as.matrix(dtm))
#length should be total number of terms
length(freq)
```

```
#create sort order (asc)
ord <- order(freq,decreasing=TRUE)
#inspect most frequently occurring terms
freq[head(ord)]
#inspect least frequently occurring terms
freq[tail(ord)]
#remove very frequent and very rare words
dtmr <-DocumentTermMatrix(docs, control=list(wordLengths=c(4, 20),
                                             bounds = list(global = c(3,27))))
freqr <- colSums(as.matrix(dtmr))
#length should be total number of terms
length(freqr)
#create sort order (asc)
ordr <- order(freqr,decreasing=TRUE)
#inspect most frequently occurring terms
freqr[head(ordr)]
#inspect least frequently occurring terms
freqr[tail(ordr)]
#list most frequent terms. Lower bound specified as second argument
findFreqTerms(dtmr,lowfreq=80)
#correlations
findAssocs(dtmr,"project",0.6)
findAssocs(dtmr,"enterprise",0.6)
findAssocs(dtmr,"system",0.6)
#histogram
wf=data.frame(term=names(freqr),occurrences=freqr)
library(ggplot2)
p <- ggplot(subset(wf, freqr>100), aes(term, occurrences))
p <- p + geom_bar(stat="identity")
p <- p + theme(axis.text.x=element_text(angle=45, hjust=1))
p
#wordcloud
library(wordcloud)
#setting the same seed each time ensures consistent look across clouds
set.seed(42)
#limit words by specifying min frequency
wordcloud(names(freqr),freqr, min.freq=70)
#...add color
wordcloud(names(freqr),freqr,min.freq=70,colors=brewer.pal(6,"Dark2"))
```