File Outline

CSV Files	
1990.2015.RE.States.Count.csv	csv written by 1990 - 2015 state/count expectancies.R and used in main R file (**Calculations.R**). Contains the run expectancy for the 24 base/out states and for a given count in the form of a data frame.
activePlayers.csv	csv with active players including name, slug, and current team. Used by server/ui to get a dropdown of current players on a specific team. Also used to pull ids for players. Needs to be updated frequently to get newly activated players. Use **activePlayers - script.R** to run through all teams for list of all players.
AL-standings.csv	csv with win/loss record for each team v. team combo. Used to get current winning percentage for each team as well as interleague record for interleague games. Updated each time **ALstandings.py** is run (which is run when **Calculations.R** is run).
deltaState.csv	csv with states and new states given a single, double, triple, or home run. Used to figure out the change in win probability if a single is hit (or double, etc.) and then used to evaluate how the batter should change the win probability. Does not need to be updated.
fields.csv	csv of header names for retrosheet play-by-play files (http://www.retrosheet.org/game.htm) used to create data file with play-by-play files from 1990 to current year. This file is then used to run **1990 - 2015 state/count expectancies.R.** Does not need to be updated (assuming format of play-by-play files does not change).
fields2.csv	csv of of header names for retrosheet game logs files used to create gamelogs.merged.csv to run **Calculations.R**. Does not need to be updated (assuming format of game logs files does not change).
gamelogs.merged.csv	csv of all game logs from 1990-2015. Used to calculate Batting Park Factors (**BPF-script.R**) and in **wpstates - script.R** file to calculate basic win probabilities. Only needs to be run at the beginning of the season with the addition of the most recent year of game logs from retrosheet — writes to csv for every park in MLB.
guts_table_2016.csv	ESPN table with league wOBA and scale for each year. Needs to be updated frequently to maintain current year's stats correctly.

	In order to update, use **guts-table-script.R** for new csv file.
NL-standings.csv	csv with win/loss record for each team v. team combo. Used to get current winning percentage for each team as well as interleague record for interleague games. Updated each time **NLstandings.py** is run (which is run when **Calculations.R** is run)
pitcher.info.csv	csv with left handed splits, right handed splits, and average splits for all current pitchers as well as pitcher names, current teams, starter or reliever, retrosheet id, retrosheet name, and debut date. Used in **Calculations.R** to get a pitcher's retrosheet id and splits.
Run.Environments.csv	csv with run environments from 1990 - 2016. Needs to be updated frequently to main current year's stats correctly. Use **RunEnvironment - script.R** to update.
splits.csv	csv with batter splits from previous year. Needs to be updated at the start of each season with the splits from the previous year. Used when a batter has had zero plate appearances and uses players who play the same defensive position to calculate a wOBA. Used in **Calculations.R** to calculate positional wOBA for regression to the mean calculation or when zero PA.
teams2016.csv	csv with current mlb teams and abbreviations for different sites used (retrosheet, espn, etc). Used to call different functions to scrape these sites for stats or to call on other csv files from those sites (i.e. retrosheet files).
wpstates.csv	csv with basic win probability for giving inning, state, and run differential (home.score - away.score). Run on data from 1990-2015 — needs to be rerun at the start of every season with new year of retrosheet data using game logs files. To update, run **wpstates-script.R** with the addition of the new data.
Python Scripts	
ALstandings.py	script to scrape the current ESPN MLB standings grid to get the winning percentage for each team (runs automatically when **Calculations.R** is run).
NLstandings.py	script to scrape the current ESPN MLB standings grid to get the winning percentage for each team (runs automatically when **Calculations.R** is run).
probabilities.py	script to run the probabilities of an event happening for a given matchup and state (single, double, triple, hr, walk) using log5 formula with batter's stats, pitcher's stats, and league average

	stats. Takes into account matchup, count, and state. State calculations are broken into: no men on, men on, in scoring position, bases loaded, in scoring position with two outs per ESPN situational stats.
batterBA.py	script to scrape current player's batting average from ESPN (runs automatically when **Calculations.R** is run).Used for log5 formula for pitcher v. batter matchup.
batterPercentages.py	script to scrape current player's percentages for hitting a single, double, triple, or home run from ESPN (runs automatically when **Calculations.R** is run).
get_league_baa.py	script to get league average batting average against used in **Calculations.R** if pitcher has not faced any batters.
leagueBA.py	script to scrape the batting average for a given league ('AL' or 'NL') from ESPN (runs automatically when **Calculations.R** is run). Used in log5 formula for pitcher v. batter matchup.
Run Environment - scrape.py	Script called on by RunEnvironment - script.R to scrape ESPN batting page to get runs/game played for MLB.
R Files	
state/count expectancies.R	R file used to get generic run expectancy matrix for the 24 base/out states and for each of the different counts. As a base file, only needs to be run once at the beginning of the season with the addition of the new retrosheet play-by-play for the given year. Takes 30-40 minutes to run.
Calculations.R	main R file called on by shinyapp server for calculations of run expectancy, win probability, and hit probabilities.
server.R	server for shinyapp (52.42.32.227/re-wp)
ui.R	ui for shinyapp (52.42.32.227/re-wp)
ss_get_result.R & set_token.R & zzz.R	stattleship files. Files are used to get the current stats using API in Calculations.R file.
activePlayers - script.R	script to get all current players with their position and slugging name used for stattleship. Data merged into a dataframe to pull ids from.
BPF-script.R	script to get updated batting park factors for all 30 teams in order to be referenced in run expectancy. Needs to be updated at the start of the new season with new game logs data from Retrosheet. *** This updates the current teams csv so that each

	team's BPF is appended to the corresponding row.
guts-table-script.R	script to pull the current ESPN table with league wOBA and scale for each year.
RunEnvironment - script.R	script to call on **Run Environment - scrape.py** to get runs scored/game played (run environment) to update Run Environment (**Run.Env**) csv. If the year is already in the csv, it will just replace that value. Otherwise, a new row will be appended to the csv.
splits - script.R	script to get pitchers' splits versus left handed and right handed batters. Writes to csv (**pitcher.info.csv**)
wpstates-script.R	script to get the basic win probabilities for a given state, run differential, and half inning. For a larger sample size than current (15 years of data), run at start of new season with last year's game logs file and play-by-play file.

Note about shiny app site : 52.42.32.227/re-wp :

To get this up and running, I used an Amazon aws EC2 instance. Since I used the most basic version, it does not support long term use of the site/many people using the site. So if you do choose to keep the tools online, it probably needs to be put on a different server.

When pushing to the server :

After pulling from github, make sure to reset the setwd by

- pwd

(copy this)

- sudo nano server.R

(delete the path to user's folder and paste new working directory)

- sudo nano RE-VA.R

(same as step above)

- sudo chmod 777 AL-standings.csv
- sudo chmod 777 NL-standings.csv

Steps to merge game log or play-by-play data :

- Need to get new data at the beginning of each season.
- Merge new season with current data (so 15 years of data in 2016 -- 16 years of data in 2017, etc) -- terminal function: cat newfile(i.e. GL2016.csv) allfiles(i.e. gamelogs.merged.csv) > gamelogs.merged.csv
- Run script with updated data
- For play-by-play data file: http://www.retrosheet.org/game.html
- For gamelogs file: http://www.retrosheet.org/gamelogs/index.html

Update Frequently	**Update at Start of Season**	**Don't Need to Update**
activePlayers.csv -- use (activePlayers - script.R)	1990.2015.RE.States.Count.csv -- use (state/count expectancies.R)	Python scripts
guts_table_2016.csv -- use (guts-table-script.R)	Gamelogs.merged.csv -- see bulleted steps above	R files (will be used to update csvs though)
pitcher.info.csv -- use (splits-script.R)	splits.csv -- use https://github.com/chadwickbureau/retrosplits/tree/master/splits once batting-byposition-yeardesired.csv is uploaded	AL-standings.csv (automatic)
Run.Environments.csv -- use (RunEnvironment - script.R)	wpstates.csv -- use (wpstates-script.R)	NL-standings.csv (automatic)
	Teams2016.csv -- use BPF-script.R to get updated batting park factors	deltaState.csv
		fields.csv
		fields2.csv