# New Season Setup Guide

Author: Caleb Sykes

Last updated: 4/12/2020

Contents

[After game is released 1](#_Toc37617150)

[ZEBRA 1](#_Toc37617151)

[Before week 0 2](#_Toc37617152)

[Determine Metrics 2](#_Toc37617153)

[Before week 1 3](#_Toc37617154)

[Event Simulator Part 1 (Data Import) 3](#_Toc37617155)

[Scouting Database 4](#_Toc37617156)

[Event Simulator Part 2 (Metrics) 5](#_Toc37617157)

[Event Simulator Part 3 (Rankings) 6](#_Toc37617158)

[Event Simulator Part 4 (User Friendly Sheets) 7](#_Toc37617159)

[Event Simulator Part 5 (Second Level Functions) 7](#_Toc37617160)

[After Week 1 and Before Week 2 8](#_Toc37617161)

[Normal Weekly Update 9](#_Toc37617162)

# About

This document contains instructions for how to build the scouting database, event simulator, and zebra parser in a new season. I plan to use this guide to make sure I don’t miss key steps when building these at the start of the new season. I also want to start building a paper trail of what I have done so that when I eventually stop maintaining these, it will at least be possible for others to pick up where I left off. If maintaining these books interests you, please reach out to me via email ([calebsyk@gmail.com](mailto:calebsyk@gmail.com)) or on CD (Caleb\_Sykes). Although I have no immediate plans to retire, it will happen eventually, and I think it would be prudent for me to train someone so my work isn’t lost.

This document is currently brand new, and likely won’t make much sense to anyone besides me. I plan to make it more readable in the future, but it was enough work just to get most of my thoughts down on paper for this year.

# After game is released

## ZEBRA

1. Download an image of the field
2. Draw lines on the downloaded field, general guideline is that zones should be between 1 and 4 robots in size.
   1. Start with lines on field elements, including field borders and markings
   2. Next, add lines that cover “invisible” penalty areas and scoring zone locations
   3. Add lines connecting corners of previously created lines, keep as many of these lines parallel to field walls as possible
   4. For remaining zones, if they are too large, break them into smaller pieces. Use corners of above lines as markers if possible.
3. Make a thread on CD asking for feedback, like this one: <https://www.chiefdelphi.com/t/2020-zebra-data-parser-zones/375721>, adjust zone definitions as necessary
4. Add IDs to each zone, move up and down adjacent zones until you get an entire side done. Zones on the opposite side of the field should have an ID of (total zones + 1) – opposite zone. Make sure to pad the zone IDs with zeros so that they can easily be sorted.
5. Give names to all zones on one half of the field, naming convention is to:
   1. Name if it’s red or blue related
   2. Say the broad area of the field it is related to
   3. If required, add a sub-area the zone is related to
   4. Add near, far, left or right descriptions to fully describe the zone. All directions will be in reference to the color specified in (a).
6. Find coordinates of all zones on one half of the field. Remember that zones must be convex, so if there are any weird intersection points to push the point in a direction to assure convexity.
7. Copy the half field zone definitions and flip them around the center coordinates of the field to obtain coordinates for the other half of the field. Likewise the names can be changed by swapping Red with Blue and vice versa.
8. Run the new zones through the first part of the validation method in the data parser. This will check for duplicate zone IDs, non-number values, and concave zones.
9. Make equivalent zones using the same indices as the zones, just with “EQ” instead of “Z”. Use own side and opposing side instead of red side and blue side. Use own instead of red and opponent instead of blue.
10. Make zone groups, make one for entire field, one for left/right and far/near, one for each penalty related zone, one for each defense related zone. Make groups for related scoring areas.
11. Make defense types, look for both general and specific types of defense.
12. Make penalties, go into the rulebook and read all of the Game Rules, marking zone-related penalties as you see them. If a specific penalty relates to multiple zones, split them up. If the penalty has foul and tech foul clauses, make a separate penalty definition for each.
13. Make auto routes (BLAH)

# Before week 0

## Determine Metrics

1. Look at the official FIRST apidocs here: <https://frcevents2.docs.apiary.io/#/reference>
2. Look at the “Match Results” section and wait for the current year’s game-specific info to be updated. This will be listed as “Response Details XXXX” where XXXX is the current year
3. Wait for TBA to match this data in their API here: <https://www.thebluealliance.com/apidocs/v3>, will be listed under “Match\_Score\_Breakdown\_XXXX” where XXXX is the current year. Verify that all fields from the apidocs are included in the tba fields
4. Copy the data import and metrics headers into a fresh workbook, also copy all fields from the tba score breakdowns into the book
5. Data import should be used to pull in raw data and perform simple linear operations:
   1. Number values should be used as is
   2. Boolean values should be used as is
   3. Strings should be split into as many Booleans as (options – 1). For example, a string field that has options for “none”, “park”, and “hang” should be split into two Boolean fields for “park” and “hang”. Unknowns can either be grouped into none or made its own Boolean entry depending on the case.
6. Add “unpenalized total points”, “win”, and “winning margin” to data import
7. Add 3 placeholders in case you forgot something and want to add it in the future
8. Using these categories, create headers for the metrics:
   1. Number values should all have calculated contributions
   2. Boolean team-specific values should have aggregated calculated contributions as well as team-specific rates
   3. There are some types of metrics that arise out of others, such as percent completions and efficiencies. For example, in 2020, define “auto inner goal efficiency” to be (auto inner goals)/(auto inner goals + auto outer goals) if (auto inner goals + auto outer goals) > 0, otherwise 0.
   4. Make sure to define penalty metrics in both directions, as it is interesting both who commits fouls and who draws fouls
   5. There are also iterative metrics that should be added such as Elo and ILSs
9. Create a Chief Delphi thread asking for feedback on these metrics. As a 2019 example: <https://www.chiefdelphi.com/t/scouting-database-event-simulator-metrics/346588>. Update metrics as necessary based on feedback

# Before week 1

## Event Simulator Part 1 (Data Import)

1. Copy the most recent simulator from the prior year
2. Request a new TBA Auth Key, here is the link for api info: <https://www.thebluealliance.com/apidocs>, and here is the link to your account dashboard: <https://www.thebluealliance.com/account>
3. Go through the first four rows of the Instructions sheet and modify as necessary
4. Unhide all sheets
5. In the JsonConverter module, note the version number. Compare to the version here: <https://github.com/VBA-tools/VBA-JSON>, if a more recent version is available, follow the instructions on that page to get the latest version.
6. In the dictionary class module, note the version number. Compare to the version number here: <https://github.com/VBA-tools/VBA-Dictionary>, if a more recent version is available, follow the instructions on that page to get the latest version. (Maybe don’t need to have the dictionary class)
7. Go to the event key sheet
   1. Update the year
   2. Update the auth key
   3. Run the function to get all events
   4. Delete the events that are not in the regular season
   5. Delete the button that calls the function
   6. Return to the Update module
8. Uncomment warning for outdated simulator and set the time to be the Thursday after week 1 competitions are completed
9. Do a find and replace for the previous year’s values and replace with the current year
10. Update the dates in the getWeek function. Use TBA as a reference: <https://www.thebluealliance.com/events>, generally Sunday works best as a cutoff date
11. Set event key in “data import” to “week0”.
12. Go line by line through the code, making updates as needed
13. Stop at the line that looks for a seed value sheet

## Scouting Database

1. Open up the most updated scouting database from the prior year
2. Save it as the database for the current year
3. Update the first 4 rows in “Instructions and FAQ” as needed as well as the FAQ
4. Go to the StartOfSeason module in vba
5. Have the current season’s event simulator open
6. In the getEventKeys method, update the eventKeySheet to match the event simulator that is currently open
7. Update the sheetNumber range to match the number of events in “Event Keys”
8. Run getEventKeys
9. Add sheets for excess events as needed
10. Delete any sheets that weren’t updated with getEventKeys, the last sheet should be for the last regular season event
11. Update the headers in the “Seed Values” sheet
12. Clear out all data in the “Seed Values” sheet
13. Go line by line through the setNamesAndFormatting function
14. Go to the importTeams function (in weekly functions)
    1. Edit the authKey
    2. Edit the current year
    3. Uncomment the one-time use section
15. Run the importTeams function
16. Run getAllTeams
17. Copy the seeds sheet into a new book
    1. Set seed values for all teams as the average of the week 0 qual matches
    2. Take Elo ratings for all teams based on the start of season Elos
    3. Find the RP1 and RP2 rates from week 0 quals matches
    4. Find the average ILS that aligns with these rates, if the RP was never achieved, use 1/(2\*(total\_matches)+1) as the rate
    5. 1 - (1 / (1 + EXP((alliance\_strength - 0.5) \* 4))) is the ILS formula, divide it by 3 to get the average team ILS.
    6. Give each team a start of season ILS for both RPs, which is average\_ILS + (SOS\_Elo – 1500)/1000
18. Copy the seeds sheet back into the scouting database
19. Update Instructions and FAQ with details on all metrics
20. Clear out the data in the world results sheet
21. Copy headings from seed values into world results
22. Update seedRow and Sheet values in setEventSeeds
23. Run setEventSeeds
24. Clear out pareto-optimal performances
25. Change sheet in getEventSummaries
26. Copy headers from seeds sheet into event summaries sheet
27. Update sheets in prepareToPublish
28. Run prepareToPublish
29. Save as xlsx file
30. Save a file to play with
31. Review the play with file for errors and resolve on xlsm if you find any
32. Save the xlsx version to GitHub and archive the previous year’s database
33. Create a CD thread announcing the update

## Event Simulator Part 2 (Metrics)

1. Find the average score and stdev of the winning margin in week0 and add these to the top of the simulateEvent method.
2. Find the first rookie in the current season, subtract a few numbers, and add that to the top of the simulateEvent method
3. Copy the all seeds sheet from the scouting database into the event sim, name as “All Seed Values WEEK 0”
4. In the simulateEvent method, change the allSeedsSheet value to be the copied sheet and comment out the other logic
5. Copy the headers from all seed values into “Seed Values Sorted” and “Seed Values”
6. Copy the headers into “Metrics” and “Metrics Sorted”
7. Copy the headers into “Calculated Contributions” and “Calculated Contributions Sorted”
8. Update the rates in the simulateEvent method to be specific for the current year
9. Go to the getBMatrix year-specific function
10. Go line by line updating as needed
11. The first column of the b matrix should be for total points, following columns should line up with the headers in “Seed Values”. It’s okay to skip rows that are blank or are ILSs/rates.
12. Update the totalCategories variable in simulateEvent to be the total number of columns in the b matrix
13. Update the isUsedCategory method. Set this function to false whenever the b matrix column is not set. That is, the columns in which predicted contributions are used
14. Copy the seed sheet headers into the red and blue prediction areas of “Data Import Sorted” (total points should be in rows 210 and 310)
15. Copy the seed sheet headers into the prediction parameters area of “Data Import Sorted” (total points should be in row 410)
16. Edit the min and max values in the prediction parameters area of “Data Import Sorted”. These should be set to the lowest and highest possible values for these categories.
17. Set the simulations to run to 0
18. Set the other settings variables to what you want for the initial upload
19. Hide all sheets except for instructions, keys, data import, and metrics
20. Comment out the resetSheet functions at the end of simulateEvent for the sheets that are hidden
21. Run the simulateEvent function in full
22. Compare metrics to the ones on TBA, Rachel Lim’s calculator, and others to look for errors
23. Update instructions and FAQ description for the sheets that are not hidden
24. Copy the category descriptions from the scouting database and put them in the instructions and FAQ sheet.
25. Update the FAQ in the instructions and FAQ sheet

## Event Simulator Part 3 (Rankings)

1. Unhide “Rankings”, “Rankings Sorted”, “Predicted Rankings”, “Predicted Rankings Sorted”, and “Settings” sheets
2. Set simulations to run in “Settings” to 100.
3. Change the inputs to the bonus RPs
4. Change the inputs to the second and third order sorts
5. Update the headers in “Rankings” and “Rankings Sorted” for the current year
6. Set “Forecast from” to “Current”, run simulateEvent and verify that rankings match up with rankings on TBA across all categories
7. Set “Forecast from” to 1 match before the last qual match, run simulateEvent and verify that the probabilities in “Predicted Rankings” all make sense.
8. Set “Forecast from” to “qm 30”, run simulateEvent and verify the probabilities in “Predicted Rankings” all make sense
9. Set “Forecast from” to “Schedule Released”, run simulateEvent and verify the probabilities in “Predicted Rankings” all make sense. Save a copy of the “Predicted Rankings” sheet
10. Set “Forecast from” to “Team List Set”, run simulateEvent. Save a copy of the “Predicted Rankings” sheet
11. Set “Forecast from” to “qm 1”, run simulateEvent. Save a copy of the “Predicted Rankings” sheet
12. Compare the saved copies of the above three sheets, all should be similar
13. Change the number of simulations in “Settings” and verify that the corresponding number of simulations are run
14. Set “Stop execution of "Update" if no new data is available” to both “Yes” and “No” and verify the behavior is as expected
15. Set “Forecast from” to “Team List Set”
16. Set “Matches per team to use for schedule if no schedule exists” to Auto. Save a copy of “Predicted Rankings”
17. Set “Matches per team to use for schedule if no schedule exists” to 1, compare predicted rankings to the saved auto sheet and you should observe less confident ranking projections
18. Set “Matches per team to use for schedule if no schedule exists” to 20, compare predicted rankings to the saved auto sheet and you should observe more confident ranking projections
19. Turn “Use advanced cell formatting in "predicted rankings"” on and off and verify that predicted rankings have sensical formatting
20. Set event key to “cust” and verify that the predicted rankings are sensical
21. Uncomment resetSheet for “Rankings”, “Predicted Rankings”, and “Metrics” at the end of simulateEvent
22. Update Instructions and FAQ for these 3 sheets

## Event Simulator Part 4 (User Friendly Sheets)

1. Unhide the “Images”, “Team Lookup”, and “Match Lookup” sheets
2. Go to “Images” tab
3. Change the file path name in cell A2 to have the current year and game name in it
4. Open up the macros for images, go to the downloadImages sub
5. Copy the authKey from simulateEvent into here
6. Find and replace all instances of the previous year number and replace them with the current year number
7. Find an event that has at least two teams which have TBA pictures
8. Test out all of the image functions on that event
9. Go to the team lookup sheet
10. Change cell N1 to be whatever the second order sort is for the current year
11. Copy the headers from “Metrics” into cells starting at cell T7
12. Update text in K8, L8, K35, and L35 to match the RPs of the current year
13. Update the RP1 and RP2 references in the code
14. Test a few different teams in team lookup (at least one with an image) and make sure everything looks right
15. Go to the “Match Lookup” sheet
16. Change the descriptions in cells C12, D12, C27, and D27 to correspond to the RPs of the current year
17. Copy category headers from “Metrics” into cells starting at cell J4
18. In the code, edit the RP1 and RP2 references
19. Try a few different matches (and at least one with a team that has an image) and make sure everything looks good
20. Try a completed match, check the number formatting in rows 14 and 15 for the results, adjust as necessary
21. At the bottom of the simulateEvent function under “Wrap up everything”, uncomment the lines related to team lookup and match lookup
22. Test out the simulateEvent function and make sure “Team Lookup” and “Match Lookup” update properly

## Event Simulator Part 5 (Second Level Functions)

1. Unhide “Overrides”, “Full-Event Graphs”, “Full-Event Data”, and “Strength of Schedule” sheets
2. Test that the schedule strength macro runs without errors
3. Change the RP name headers in “Overrides” for the current year
4. Update the RP references to data import in the importPredictions function
5. Test out the importPredictions function
6. Run the simulateEvent function without overrides
7. Create a few very obvious overrides and simulate the event with them, making sure the results look roughly like what you would expect
8. Copy the headers from “Metrics” into “Full-Event Data” starting in cell OM99
9. Copy the headers from “Metrics” into “Full-Event Graphs” starting in cell PF5
10. In the “Full-Event Graphs” sheet, run Re-generate Full Event Data
11. Test a few different teams and graph settings
12. Hide “Full-Event Data” sheet
13. Update the descriptions for these sheets in “Instructions and FAQ”
14. You’re finally done for this week!

# After Week 1 and Before Week 2

1. Create a new book called “all week 1 results”
2. In the simulator, set “simulations to run” in settings to be 0
3. Set “Forecast from:” to “Current”
4. Simulate each week 1 event, copy the results from data import into the same page of the new book, going down 200 cells for each copy
5. Find the average score and stdev of the winning margin over all qual and playoff matches
6. Go into the simulateEvent method and change the stdev and average variables for the current year at the top of the function
7. Find the rates at which the bonus RPs were achieved (make sure it’s quals only)
8. Open the scouting database
9. In the importCCs function, change the simulator name and the sheets containing week 1 events and run this function
10. Change the sheet variable in setWorldResults and run that function
11. Copy the world results sheet into a new book
12. Find the average and standard deviation for each calculated contribution category
13. Find the average for all rookie teams in each category
14. Copy the week 0 seed values sheet from the scouting database into a new book called “Official Week 0 Seeds”
15. Open up the previous year’s scouting database and copy the seed values sheet into the “Official Week 0 Seeds” book
16. In the “Official Week 0 Seeds” book, name the two sheets something different so you don’t get confused
17. Delete all data to the right of Elos in the current year seed value sheet
18. Add a column at the end of the current year seed value sheet called “seed OPR”, fill this column with a vlookup search from the “total Points” category of the previous year’s seed values sheet
19. Find the average and stdev of the seed OPRs
20. Make a column next to seed OPRs called “normalized”, fill the data in this column with ((seedOPR) – (average\_seedOPR))\*0.9/(stdev\_seedOPR). The 0.9 was found according to this analysis: <https://www.chiefdelphi.com/t/paper-4536-scouting-database-2017/157268/3>
21. Fill the data with the following formula: =IF(hasPreviousYearSeed,normalized\_previousYearOPR\*categorySTDEV+categoryAvg,rookieAvg)
22. For the penalty points, subtract instead of add since lower penalties are better
23. Set the seed strengths to be 1 for rookies/new veterans and 2 for returning teams
24. Give each team a start of season ILS for both RPs, which is average\_ILS + (SOS\_Elo – 1500)/1000
25. Clean up this sheet, remove formulas
26. Replace the seed values sheet in the scouting database with this one
27. Replace the seed values sheet in the event simulator with this one
28. You’re all set now for a normal update procedure!

# Normal Weekly Update

1. Open up the scouting database
2. Change the sheetIndex range in clearAndSetFormatting to include all incomplete events and run this function
3. In importCCs, Change the sheetIndex range to cover all events for the current week. Update the simulatorName to be the most recent event simulator. Run this function.
4. In importTeams, change the sheetIndex range to cover all upcoming events
5. In setWorldResults, change the sheetIndex range to cover all events in the current week. Make sure the “Seed Values” sheet is sorted by team number ascending. Run this function
6. In the setSeeds function, make sure world results is sorted first by start date newest to oldest, then second by team number ascending, run this function
7. In the setEventSeeds function, change the sheetIndex range to cover all upcoming events. Run this function
8. Run findParetoOptimalPerformances
9. Run getEventSummaries
10. Go to prepareToPublish, change the sheetIndex for upcoming events. Run this function