# New Season Setup Guide

Author: Caleb Sykes

Last updated: 3/3/2022

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 (skipped for 2022)

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://frc-api-docs.firstinspires.org/>
2. Look at the “Match Results” section, go to “GET Score Details” and on the right pane use the drop down menu titled “20XX Score Details” to select the current year’s game-specific info to be updated.

Graphical user interface, text, application, chat or text message, website

Description automatically generated

1. Wait for TBA to match this data in their API here: <https://www.thebluealliance.com/apidocs/v3>, will be listed under “Match\_Score\_Breakdown\_20XX” where 20XX is the current year. Verify that all fields from the apidocs are included in the tba fields
2. Copy the data import and metrics headers into a fresh workbook, also copy all fields from the tba score breakdowns into the book
3. 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.
4. Add “unpenalized total points”, “win”, and “winning margin” to data import
5. Add 3 placeholders in case you forgot something and want to add it in the future
6. 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
7. 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>. Update this key in “Local Saves” row 2 column 1, make sure to remove leading and trailing whitespace.
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. If trying to support macs: 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. Dictionary class module is required to support macs using VBA-JSON
7. Go to the event key sheet
   1. Update the year in the request (line
   2. Run the function to get all events
   3. Delete the events that are not in the regular season
   4. Delete the button that calls the function
8. Return to the Update module
9. Uncomment warning for outdated simulator and set the time to be the Thursday after week 1 competitions are completed
10. Do a find and replace for the previous year’s values and replace with the current year
11. 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
12. Set event key in “data import” to “week0”.
13. Go line by line through the code (particularly the formatMatch method in YearSpecificFunctions module), making updates as needed
14. Stop at the line that looks for a seed value sheet

## Scouting Database

1. Open up the most updated xlsm scouting database from the prior year
2. Save it as the database for the current year
3. 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>. Update this key in “AuthKey” row 2 column 1, make sure to remove leading and trailing whitespace.
4. Update the first 3 rows in “Instructions and FAQ” as needed as well as the FAQ
5. Go to the StartOfSeason module in vba
6. Have the current season’s event simulator open
7. In the getEventKeys method
   1. update the eventKeySheet to match the event simulator that is currently open
   2. Update the sheetNumber range to match the number of events in “Event Keys”
   3. Run getEventKeys
8. Add sheets for excess events as needed
9. Delete any sheets that weren’t updated with getEventKeys, the last sheet should be for the last regular season event
10. Update the headers in the “Seed Values” sheet
11. Clear out all data in the “Seed Values” sheet
12. Go line by line through the setNamesAndFormatting function
    1. Change the range on the AutoFilter
    2. If you get a Run-time error ‘1004’ saying “That name is already taken. Try a different one like this, then click debug:

Graphical user interface, text, application

Description automatically generated

* 1. Look at the event key of the sheet it is trying to set and search TBA for that event name
  2. This event has the same short name as another event’s short name (or at least the first 31 characters are the same (for example, there might be two events with short name “St Joseph”, one in Indiana and one in Michigan)
  3. Search all tab names you have already set and change the first name to be unique.
  4. Progress one line in debug and then change the name you just set to be unique.
  5. Continue the function

1. Go to the importTeams function (in weekly functions)
   1. Edit the current year
   2. Uncomment the one-time use section
2. Run the importTeams function
3. Go to getAllTeams in StartOfSeason module
   1. Update sheetIndex to cover all events
   2. Run this function
4. Copy the seeds sheet into a new book
   1. Set seed values for all teams as the average of the week 0 qual matches divided by 3
   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
5. Copy the seeds sheet back into the scouting database
6. Update Instructions and FAQ with details on all metrics
7. Clear out the data in the world results sheet
8. Copy headings from seed values into world results
9. in setEventSeeds in WeeklyFunctions module
   1. Update seedRow and
   2. Update Sheet values
   3. Update the columns to be copied
   4. Uncomment the line to paste column widths
   5. Run setEventSeeds
   6. Comment the line to paste column widths
10. Clear out pareto-optimal performances
11. Copy headers from seeds sheet into event summaries sheet
12. In getEventSummaries in Weekly Functions module
    1. Change sheet
    2. Run getEventSummaries
13. In prepareToPublish in weekly Functions module
    1. Update sheets
    2. Update column formatting
    3. Run prepareToPublish
14. Save as xlsx file
15. Save a file to play with
16. Review the play with file for errors and resolve on xlsm if you find any
17. Save the xlsx version to GitHub and archive the previous year’s database
18. 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 seed values 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. Make sure auto filter covers all values in “Seed Values”
7. Copy the headers into “Metrics” and “Metrics Sorted”
8. Make sure auto filter covers all values in “Metrics”
9. Copy the headers into “Calculated Contributions” and “Calculated Contributions Sorted”
10. Make sure auto filter covers all values in “Calculated Contributions”
11. Update the rates in the simulateEvent method to be specific for the current year
12. Update the totalCategories variable in simulateEvent to be the total number of columns in the b matrix
13. Go to the getBMatrix year-specific function
14. Go line by line updating as needed
15. 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.
16. Update the unusedCategories list in “2022 Parameters” sheet.
17. 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)
18. Copy the seed sheet headers into the prediction parameters area of “Data Import Sorted” (total points should be in row 410)
19. 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.
20. Set the simulations to run to 0
21. Set the other settings variables to what you want for the initial upload
22. Hide all sheets except for instructions, keys, data import, settings, and metrics
23. Comment out the resetSheet functions at the end of simulateEvent for the sheets that are hidden
24. Run the simulateEvent function in full
25. Compare metrics to the ones on TBA, Rachel Lim’s calculator, and others to look for errors
26. Update instructions and FAQ description for the sheets that are not hidden
27. Copy the category descriptions from the scouting database and put them in the instructions and FAQ sheet.
28. 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 the Event key to an event that doesn’t have a schedule yet (week 2 or later)
16. Set “Forecast from” to “Team List Set”
17. Set “Matches per team to use for schedule if no schedule exists” to Auto. Save a copy of “Predicted Rankings”
18. 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
19. Set “Matches per team to use for schedule if no schedule exists” to 14, compare predicted rankings to the saved auto sheet and you should observe more confident ranking projections
20. Turn “Use advanced cell formatting in "predicted rankings"” on and off and verify that predicted rankings have sensical formatting
21. Set event key to “cust” and verify that the predicted rankings are sensical
22. Uncomment resetSheet for “Rankings” and “Predicted Rankings” at the end of simulateEvent
23. 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 the Images sheet, go to the downloadImages sub
   1. Find and replace all instances of the previous year number and replace them with the current year number
5. Find an event that has at least two teams which have TBA pictures
6. Test out all of the image functions on that event
   1. Test “Download Images”
   2. Test “Create Folder”
   3. Test “Update Images”
7. Go to the “Team Lookup” sheet
   1. Change cell N1 to be whatever the second order sort is for the current year
   2. Copy the headers from “Metrics” into cells starting at cell T7
   3. Update text in K8, L8, K35, and L35 to match the RPs of the current year
   4. Update the RP1 and RP2 references in the code
   5. Test a few different teams (at least one with an image) and make sure everything looks right. Make sure to test with matches still to be played, either by using an ongoing event or setting “Forecast From” in “Data Import” to a mid-event qual match.
8. Go to the “Match Lookup” sheet
   1. Change the descriptions in cells C12, D12, C27, and D27 to correspond to the RPs of the current year
   2. Copy category headers from “Metrics” into cells starting at cell J4
   3. In the code, edit the RP1 and RP2 references
   4. Try a few different matches (and at least one with a team that has an image) and make sure everything looks good
   5. Try a completed match, check the number formatting in rows 14 and 15 for the results, adjust as necessary
9. At the bottom of the simulateEvent function under “Wrap up everything”, uncomment the lines related to team lookup, match lookup, and images
10. Test out the simulateEvent function and make sure “Team Lookup” and “Match Lookup” update properly
11. Update the descriptions for these sheets in “Instructions and FAQ”

## 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. Go to “Overrides
   1. Change the RP name headers
   2. Update the RP references to data import in the importPredictions function
   3. Test out the importPredictions function
4. Run the simulateEvent function without overrides
5. Create a few very obvious overrides and simulate the event with them, making sure the results look roughly like what you would expect
6. Copy the headers from “Metrics” into “Full-Event Data” starting in cell OM99
7. Copy the headers from “Metrics” into “Full-Event Graphs” starting in cell PF5
8. In the “Full-Event Graphs” sheet, run Re-generate Full Event Data
9. Test a few different teams and graph settings
10. Hide “Full-Event Data” sheet
11. At the bottom of the simulateEvent function under “Wrap up everything”, uncomment the lines related to strength of schedule, overrides, and full-event graphs
12. Update the descriptions for these sheets in “Instructions and FAQ”
13. 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 eventSimulator, run dumpWeek1Results in the StartOfSeason module
3. Find the average score and stdev of the winning margin over all qual and playoff matches
4. Go into the simulateEvent method and change the stdev and average variables for the current year at the top of the function
5. Find the rates at which the bonus RPs were achieved (make sure it’s quals only)
6. Open the scouting database
7. In the importCCs function, change the simulator name and the sheets containing week 1 events and run this function
8. Change the sheet variable in setWorldResults and run that function
9. Copy the world results sheet into a new book
10. Find the average and standard deviation for each calculated contribution category
11. Find the average for all rookie teams in each category
12. Copy the week 0 seed values sheet from the scouting database into a new book called “Official Week 0 Seeds”
13. Open up the previous year’s scouting database and copy the seed values sheet into the “Official Week 0 Seeds” book
14. In the “Official Week 0 Seeds” book, name the two sheets something different so you don’t get confused
15. Delete all data to the right of Elos in the current year seed value sheet
16. 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
17. Find the average and stdev of the seed OPRs
18. 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>
19. Fill the data with the following formula: =IF(hasPreviousYearSeed,normalized\_previousYearOPR\*categorySTDEV+categoryAvg,rookieAvg)
20. For the penalty points, subtract instead of add since lower penalties are better
21. Set the seed strengths to be 1 for rookies/new veterans and 2 for returning teams
22. Give each team a start of season ILS for both RPs, which is average\_ILS + (SOS\_Elo – 1500)/1000
23. Clean up this sheet, remove formulas
24. Replace the seed values sheet in the scouting database with this one
25. Replace the seed values sheet in the event simulator with this one
26. In the importTeams function in the scouting database, comment out the section that is only run at the beginning of the season
27. You’re all set now for a normal update procedure!

# Normal Weekly Update

1. Open up the most recent xlsm scouting database
2. Save it with a new name as the current version number: XXXX.Y.Z where XXXX is the current year, Y is the event competition week that data is being uploaded for, and Z is 0 for the first update for the week, and 1 higher than the previous release number if there has already been an update this week.
3. Open up the most recent event simulator
4. Open the weeklyUpdate module:
   1. in the clearAndSetFormatting function:
      1. Change the sheetIndex range to include all incomplete events
      2. run this function.
   2. In importCCs,
      1. Change the sheetIndex range to cover all events for the current week.
      2. Update the simulatorName to be the name of the open most recent event simulator.
      3. Run this function.
   3. In importTeams,
      1. change the sheetIndex range to cover all upcoming events
      2. Run this function
   4. In setWorldResults,
      1. change the sheetIndex range to cover all events in the current week.
      2. Run this function
   5. Run the setSeeds function
   6. In the setEventSeeds function,
      1. change the sheetIndex range to cover all upcoming events.
      2. Run this function
   7. Run findParetoOptimalPerformances
   8. Run getEventSummaries
5. Update Instructions and FAQ as appropriate
6. Go to prepareToPublish,
   1. change the sheetIndex for upcoming events.
   2. Run this function
7. In the master branch:
   1. Save file and place it in the github most\_recent\_xlsm folder
   2. Save the file as an xlsx and place it at the top level of the github folder
   3. Move existing xlsx database into the archive folder
   4. Push master changes to github
8. In the most\_recent\_release branch
   1. Save the new xlsx database to the top-level folder
   2. Delete the old database at this location
   3. Push most\_recent\_release changes to github
9. On the SykesScoutingDatabase on Github:
   1. Click on “Releases”
   2. Click “Draft a new release”
   3. Click “Choose a tag” and enter the version number of the database (vXXXX.Y.Z as in step 2 above)
   4. Select “Create a new tag: vXXXX.Y.Z on publish”
   5. Make sure branch is set to “most\_recent\_release”
   6. Fill in the title and description fields
   7. Click “Publish release”
   8. Go back to the “releases” page.
   9. Under the most recent release, right click on “Source code (zip)” and copy the link
10. Create a new CD post under the scouting database thread and paste the link as well as the description.
11. Copy the “Seed Values” sheet from the scouting database into the event simulator
12. Rename this sheet to be “Seed Values m-d” where m is the month and d is the day prior to any competitions not included in the seeds sheet.
13. Hide this sheet in the event simulator
14. In the simulateEvent module of the event simulator, add logic for allSeedsSheet following the formula of the other seeds.
15. SKIP THIS STEP FOR NOW: Set "Stop execution of "Update" if no new data is available” to "Yes"
16. In “Data Import”
    1. Set "Forecast from" to "Current"
17. Update Instructions and FAQ as appropriate
18. Make sure correct sheets are hidden
19. Change the date on the fourth line of simulateEvent module to be one week after the current date
20. In the master branch:
    1. Save the file at the top level of the github folder
    2. Move existing simulator into the archive folder
    3. Push master changes to github
21. In the most\_recent\_release branch
    1. Save the new simulator to the top-level folder
    2. Delete the old simulator at this location
    3. Push most\_recent\_release changes to github
22. On the SykesEventSimulator on Github:
    1. Click on “Releases”
    2. Click “Draft a new release”
    3. Click “Choose a tag” and enter the version number of the database (vXXXX.Y.Z as in step 2 above)
    4. Select “Create a new tag: vXXXX.Y.Z on publish”
    5. Make sure branch is set to “most\_recent\_release”
    6. Fill in the title and description fields
    7. Click “Publish release”
    8. Go back to the “releases” page.
    9. Under the most recent release, right click on “Source code (zip)” and copy the link
23. Create a new CD post under the scouting database thread and paste the link as well as the description.
24. You are done for the week!