# Title Slide

Good afternoon and welcome back to EDirect for PubMed! Today is Part Five: Developing and Building Scripts.

My name is Mike Davidson, and I’m a librarian at the National Library of Medicine in Bethesda, Maryland.

# EDirect for PubMed Agenda

This is the final of five sessions in the EDirect for PubMed series. (Thanks for sticking with us!)

Today is all about putting all of the pieces we’ve looked at so far together, and talking about how we can develop and build solutions to real world problems.

# Today’s Agenda

We’re going to start with a quick recap of Part Four

We’re then going to spend a few minutes discussing how we can put all of the tools we’ve learned about to use, and go over some methods for developing solutions.

We’ll finish off with a deep dive into a few real-world case studies, to look at the whole process from beginning to end.

# Recap of Part Four

Hopefully you remember that last week, we spent most of the class talking about Conditional arguments.

We used -if to limit our output, based on if a -pattern or -block met a certain condition.

We used -if by itself to limit based on the presence of a particular element or attribute.

We used -if with -equals to limit based on whether an element or attributed equaled a particular string.

And we used -if with -contains to limit based on whether an element contained a certain string.

# Recap of Part Four (cont’d)

We also talked about the two Boolean conditional arguments: -or requires that at least one of the conditions we specify must be met, and -and requires that both conditions must be met.

We did a quick discussion of -position, which lets us include a specific -block, based on its position in a -pattern, whether it is the first block, the second block, the last block, etc.

And finally, we talked about one argument that’s not Conditional: -def, which lets us specify a placeholder to appear where there would otherwise be blank cells in our output table.

# Questions from last class? Homework?

Before we get into today’s class, does anyone have any questions about anything we discussed in any of the previous sessions? Or any questions about the homework?

**[PAUSE FOR QUESTIONS]**

# We have all the pieces… (ANIMATED)

Over the four sessions, we’ve spent a lot of time introducing you to new tools and techniques. Some of these tools are new ways of doing things we can already do in PubMed, **[CLICK]** like esearch, which searches a database, **[CLICK]** and efetch, which retrieves full records in XML.

Some of these tools allow us to do new things that we can’t do in PubMed,**[CLICK]** like xtract, which can extract data from XML and arrange it in a table.

We now have all these tools that let us answer advanced PubMed questions, but **[CLICK]** how do we actually **do** it? How can we put these tools to use? Learning how to use these tools is only part of the challenge, you also need to figure out when to use which tools and in what order. Basically, we need to learn how to be software developers!

A lot of this comes just through experience, practice, and experimentation. However, I can offer you some advice based on the hours and hours I’ve spent trying to build scripts to do specific tasks.

# Strategies for Developing a Script

Now, I’ll walk you through my process, but these suggestions may not work for everyone. And that’s okay. If you find a different way of thinking about this process that works for you, go with that. There’s no one right answer. That said, here are some things I think about when I’m trying to develop a solution in EDirect.

# 1. Identify your goal (ANIMATED)

Before starting to build a script, it can be helpful to think about your goal, and make sure you have a good idea of what you're trying to accomplish.

**[CLICK]** The first thing you want to do is identify your input. In other words, what do you already know? Maybe you have a PubMed search strategy. Maybe you have a list of PMIDs for PubMed records that you want to retrieve and analyze. Maybe you already have a bunch of PubMed records in XML. What type of input you have is going to determine how you build the first steps of your script.

**[CLICK]** Next, identify your output: What is it that you want to know? Do you want a list of PMIDs? Do you want specific data from within a PubMed record? Which fields specifically are you trying to output?

**[CLICK]** Then, identify your output format: How do you want your data arranged? Is XML good enough? Do you want a table instead? If so, what order should the columns be in? How should the columns be separated? Do you want your output saved to a file? What type of file? These details will help you refine and polish your script, to get the results in the exact format you need.

# 2. Choose your tool

Now, we just spent four sessions teaching you EDirect, but you want to make sure that EDirect is actually the right tool for the job. I know when I learn about a new tool, I am tempted to go out looking for projects to use it on. It’s like the old expression: “If you have a hammer, everything looks like a nail.” You want to make sure the tool you choose serves the goal of the project, not the other way around.

EDirect can be very useful, but it isn’t the always the most practical solution to a problem. If you can do the job faster by just going to PubMed and getting your answer that way, then you may want to consider that option instead. Likewise, if your project requires access to an incredibly large amount of PubMed data (like, for example, all of PubMed), you may find EDirect is not the best choice.

# Working with ALL of PubMed

To avoid overloading the E-utilities servers, NCBI has set forth some usage guidelines which, among other things, ask users to limit high-volume usage to off-peak hours. You can find more about those usage guidelines on NCBI’s website (there’s a URL for this page in your handout, though the guidelines are a little dense, so I’d suggest you wait until after class to look at them).

Additionally, incredibly high-volume scripts take a lot of time to run, so might not be terribly practical.

If your project requires access to all of PubMed, I would instead direct you to the NLM Data Distribution program, which provides bulk downloads of MEDLINE records in XML. (Again, there’s a URL in the handout.)

Until recently, we used to mention the NLM Data Distribution as an alternative to folks who wanted to get work with the entirety of PubMed, but we knew it wasn’t practical for everyone.

Many of our Data Distribution customers are companies or researchers who have the infrastructure to take these bulk downloads, pull out the XML, and file the records into their own homegrown databases so they can work with them. And many of them have teams of developers to make sure all of that works correctly. Most EDirect users don’t have these resources.

# Get the best of both worlds?

However, starting with EDirect version 8.00, there’s a new feature that can help you get the best of both worlds. It takes the flexibility of EDirect, and puts it to work on the entire 28 million records of PubMed.

This new feature lets you build a local copy of PubMed on your own drive, so you don’t have to wait for E-utilities to fetch the records every time. You can still use esearch to search through the live PubMed database, but rather than fetching the XML with E-utilities, you fetch it from your local copy. This can speed up your retrieval exponentially, especially when you’re trying fetch tens or hundreds of thousands (or even millions) of records.

There are a few caveats with this process.

First, in order to actually see any speed benefit, you’ll need to have an external solid state drive, and you’ll want it to be at least 500 GB.

Second, building the local copy takes some time. Depending on your system, it could take anywhere from six to over thirty hours the first time you build it.

As you can see, this might be a useful option for some projects where you’re analyzing huge amounts of data, but, for many projects where you’re retrieving less than ten thousand records, you’re probably better off using EDirect the old fashioned way, given the startup costs.

However, if you’re interested in pursuing this, there are some links in the handout, and I’ll be happy to address questions on it at the end of today’s class, or on Thursday during office hours.

There is one other way to get the best of both worlds: You may also want to remember that xtract works on any XML (using the –input argument), so even if you don’t get your XML from efetch, but instead get it from NLM Data Distribution, from using the PubMed “Send to: File” option to download XML from the web version of PubMed, or from some other source you can still use xtract.

# 3. Understand the data

While we’re talking about data, you want to make sure that, just because we’re using an exciting new tool, you don’t forget or ignore everything you know about PubMed data.

The solutions you build with EDirect are only going to be as good as the data that you’re retrieving, and as good as your knowledge of that data.

Also, keep in mind what data is and isn’t available in PubMed. Remember, PubMed doesn’t contain full text, only citations to data. Additionally, not all fields are populated for all records. For example, MeSH headings are only attached to MEDLINE records. MEDLINE records make up the bulk of PubMed, but because the indexing process takes time, you’ll find that a lot of very recent publications don’t have MeSH headings, because they haven’t yet been fully indexed.

Even when data is available, you’ll want to think about what that data looks like. Author affiliation data is supplied by the publisher as free text, and is not controlled or QC’d by NLM. Likewise, there can sometimes be some inconsistences in grant identifiers.

Understanding the data will help you figure out what’s possible, and will also highlight issues that you might need to address.

If you want to improve your PubMed searching skills, NLM and the National Network of Libraries of Medicine offer a variety of tutorials and training opportunities.

If you want to get more familiar with the data, remember the PubMed XML documentation we looked at in the second session, as that’s a good place to start.

# 4. Decide how much to automate

One of the big advantages of EDirect is that it lets you automate repetitive and time-consuming jobs. However, developing an EDirect script can also take time.

As we already mentioned, you want to consider whether the efficiency you gain is worth the time investment. Of course, by working with EDirect, you get the added benefit of learning more about how it works and getting better at it, so you can factor that into your evaluation, too. But one area where you really want to make sure you consider how much your investment is worth is when figuring out when to stop.

As we’ve said many times with EDirect, there are almost always multiple ways of doing what you want to do. There is almost always a way to accomplish 100% of your goal in a single script. However, there are usually also ways of accomplishing 90%, 75% or 50% of your goal in a single script, and doing the remaining 10%, 25% or 50% either manually, or using a different tool (like Microsoft Excel or other data analysis and visualization tools). You’ll have to decide whether the additional time and effort it will take to get your script from 90% to 100% is more or less efficient than doing the remaining 10% manually.

This also brings up another point, which is that sometimes a solution will require actual human judgement, for evaluating results, determining relevance, or dealing with some of the data issues we were talking about before.

Ideally, you can automate the repetitive parts which will free up more time for you to perform the tasks that require a human. Figuring out if and where a human fits should be something you think about during your development process.

Before we go on, I really want to underline the “90-10” concept, as it can be a very helpful way to think about automation.

One of the main points of automation is to save you time, so if your 90% solution saves you time, then you might not need to go any farther, even if your solution isn’t 100% perfect. Basically, don’t let the perfect be the enemy of the good.

# 5. Build one step at a time

I’ve mentioned this many times before, but it’s worth reiterating here.You’ll be better able to spot issues in your development if you create commands one at a time, testing and troubleshooting as you go.

# Case Study

To show you more of what I mean, we’re going to spend the rest of the session looking at a case study. We’re not just going to break down the code of the finished product. Instead, we’re going to develop this solution from the ground up, looking at how we choose, build, and test each command.

# Case Study: Our Goal

First, let’s identify our goal.

We want a list of articles about breast cancer that were published in the last year, and are linked to entries in ClinicalTrials.gov. For each article, we need to know the PMID, NCT Number(s), first author, and the journal it was published in.

# Case Study: Identify your input

Basically, our input will be a PubMed search string: “breast cancer AND clinicaltrials.gov[si]”. We’re looking for breast cancer as a subject, and for clinicaltrials.gov in the supplemental information field.

We also know that we want to limit our search by date, to articles published between March 1, 2017 and February 28, 2018.

Between the search string and the date restriction, we know we’ll be starting with an esearch command.

# Case Study: Identify your output

We have four elements that we want in our output.

For PMID, we’ll want to specify the correct PMID element using Parent/Child construction.

The XML element for NCT Number is “AccessionNumber”, which is child of the DataBank element. However, we only want to use the NCT numbers, not other Accession Numbers. We’ll have to make sure we’re only using AccessionNumbers from DataBanks where the DataBank name is ClinicalTrials.gov

# Case Study: Identify your output (cont’d)

For first author, we want the LastName and Initials, but only for the first author. This is going to be a good opportunity to use that –position argument we were talking about earlier.

Finally, we want the journal title abbreviation, which you might remember, is ISOAbbreviation.

# Case Study: Identify your format

Now let’s talk about our format.

Each article is going to have its own row, and we’ll have four columns, but what order do we want our columns to be in? I’m going to do PMID, then Journal, then First Author, then NCT number.

We’ll put NCT Number last because it’s repeatable, and we don’t want it throw off the alignment of our other data.

We can separate the columns by tabs, and we’ll separate multiple NCT numbers, if a record has more than one, by pipes.

Finally, we want to save this to a file, so we can open it in Excel and read, visualize or analyze our data more easily.

# Case Study: Time to build!

(SWITCH TO CYGWIN)

## esearch

We’ll start off with the easiest part: building an esearch line. As you might remember from our first class, esearch commands are pretty simple to write, quick to run, and easy to evaluate whether they’re working.

We’ll start with just the search string:

esearch -db pubmed -query "breast cancer AND clinicaltrials.gov[si]"

(EXECUTE)

Now let’s compare the results to running the same search in PubMed, just to make sure.

(SWITCH TO PUBMED. Search “breast cancer AND clinicaltrials.gov[si]”)

Our number of results are the same, so we know our esearch works.

(SWITCH TO CYGWIN)

Then we can add our date filter:

esearch -db pubmed -query "breast cancer AND clinicaltrials.gov[si]" \

-datetype PDAT -mindate 2017/03/01 -maxdate 2018/02/28

We could also include our date restriction inside our search query, but I’m going to keep it separate, as it makes it easier to make adjustments to the date restrictions without affecting the rest of the search string.

(EXECUTE)

Now, for testing purposes, let’s scale our search back to an even smaller time frame.

esearch -db pubmed -query "breast cancer AND clinicaltrials.gov[si]" \

-datetype PDAT -mindate 2018/01/01 -maxdate 2018/02/28

(EXECUTE)

## efetch

Now, let’s put an efetch command after that to grab a few PMIDs for testing purposes:

esearch -db pubmed -query "breast cancer AND clinicaltrials.gov[si]" \

- datetype PDAT -mindate 2018/01/01 -maxdate 2018/02/28 | \

efetch –format uid

(EXECUTE)

I’ll take a couple of these PMIDs and put them into an efetch with the –format xml, so I can get some good raw data to work on when building my xtract command:

efetch -db pubmed -id 29172605,29158011,29136523,29045554,29045543,28918548,28741175,28702218 -format xml

(EXECUTE)

## xtract basics

Now let’s start building our xtract.

We know our –pattern is going to be PubmedArticle. Our first two elements are also easy:

efetch -db pubmed -id 29172605,29158011,29136523,29045554,29045543,28918548,28741175,28702218 -format xml | \

xtract -pattern PubmedArticle -element MedlineCitation/PMID ISOAbbreviation

Let’s test that part of our xtract.

(EXECUTE)

## xtract First Author

Then we need our first author.

We’ll use –block Author, so we can better explore our author information, and we can limit our data to just the first –block using –position first, or –position 1. Our –element argument will be LastName,Initials, and we’ll include a –sep “ “ to give us that nice formatting.

efetch -db pubmed -id 29172605,29158011,29136523,29045554,29045543,28918548,28741175,28702218 -format xml | \

xtract -pattern PubmedArticle -element MedlineCitation/PMID ISOAbbreviation \

-block Author -position 1 -sep " " -element LastName,Initials

Let’s test that part of our xtract.

(EXECUTE)

## xtract NCT numbers

Now let’s do the NCT number.

We know that we need to go into the –block DataBank, and the –element is going to be AccessionNumber. We know that we want to separate multiple NCT numbers with “|”, so we’ll use –sep “|”.

However, we only want to include the AccessionNumber for ClinicalTrials.gov records. We’ll need an –if to weed out the DataBank –blocks that aren’t for ClinicalTrials.gov. In other words, we only want to include the block **if** DataBankName **equals** ClinicalTrials.gov.

## Putting the xtract together

Putting our whole xtract together, we get:

efetch -db pubmed -id 29172605,29158011,29136523,29045554,29045543,28918548,28741175,28702218 -format xml | \

xtract -pattern PubmedArticle -element MedlineCitation/PMID ISOAbbreviation \

-block Author -position 1 -sep " " -element LastName,Initials \

-block DataBank -if DataBankName -equals ClinicalTrials.gov \

-sep "|" -element AccessionNumber

(EXECUTE)

## Putting the script together

Now let’s put the whole script together, with our esearch line that we made at the beginning, our efetch –format xml, and our xtract line. And don’t foget we want to save this to a file, so we’ll add on our “>” at the end to redirect to a file:

esearch -db pubmed -query "breast cancer AND clinicaltrials.gov[si]" \

-datetype PDAT -mindate 2017/03/01 -maxdate 2018/02/28 | \

efetch -format xml | \

xtract -pattern PubmedArticle -element MedlineCitation/PMID ISOAbbreviation \

-block Author -position 1 -sep " " -element LastName,Initials \

-block DataBank -if DataBankName -equals ClinicalTrials.gov \

-sep "|" -element AccessionNumber > clinicaltrials.txt

(EXECUTE)

(ONCE COMPLETE:) Now let’s open up our file in Excel to see our results!

(DEMO IN EXCEL: SHOW OUTPUT FILE)

And we’ve accomplished our goal: we have the information we need, but only the information we need, in exactly the format we wanted.

# Next steps… (ANIMATED)

That’s about all I have for you today. Thanks for sticking with us over the past five classes.

I do want to say, though, that if you feel like you still don’t really know how to use EDirect after five 75-minute sessions, I don’t blame you. It certainly took me a lot longer to learn this stuff. There’s a lot to absorb here, and you’re probably only going to be able to really learn it by actually working with EDirect yourself.

I hope we were able to show you why EDirect is so useful, and introduce you to some of the tools that can help you put it to use, but I hope you will take time to review the sample code on the website, look at the recordings, poke through some of the documentation and experiment with some of the exercises and demonstrations.

It does take time and practice to really master these tools, so please: give the tools a try. Experiment. Play around with some of the commands we’ve talked about. Get comfortable in the command line environment.

We have one final assignment for this class, which I’ll be telling you about in a minute. It gives you the opportunity to solve some real-world problems with some of the stuff you’ve learned over the past few weeks. I encourage you to use it as another opportunity to roll up your sleeves and really dig in to EDirect.

And finally, keep your eyes open for the next time a project that would be a good fit for EDirect comes along.

In the meantime, I have one more resource to show you, that’s still pretty new, but you might find useful. **[CLICK]** NCBI has released an EDirect “cookbook”: basically a list of premade scripts that can accomplish simple tasks. Since looking at and modifying other people’s examples is the best way to start working with EDirect, it’s a pretty good resource to look at.

(SWITCH TO BROWSER: https://ncbi-hackathons.github.io/EDirectCookbook/)

The cookbook has been published via GitHub, which is an online repository for people to share and work on code together. If you haven’t ever used GitHub, don’t worry…you don’t need to know anything about it to use the cookbook. Simply go to the site (which Sarah will put in the chatbox, and which is linked from the Class Materials page), and copy and paste any of the scripts that are up there.

For now, most of the scripts are more focused on the molecular biology and genetics databases, but there are a few PubMed scripts up there, including some of the examples from this class, and we’re trying to get a few more up there, too.

But the nice thing about GitHub is that if you need a script that does a particular thing, but there isn’t one here already, you can use this Issues tab up here to request a script. And on the flip side, if you have a script that you want to share, you can do that too.

Like I said, this is very new, and they’re still working out some of the kinks, but hopefully this will let us develop a sort of EDirect community of practice. (BACK TO SLIDES)

As we’ve mentioned before, make sure you subscribe to the utilities-announce listserv. We don’t have any other classes scheduled right now, but as soon as new offerings are available, we’ll announce them on the website and through the listserv. And for those of you who are curious about MLA CE credit, we will send you information about how to claim that credit…as soon as you finish your final assignment.

# Final Assignment

The final assignment isn’t that much more complicated than the homework questions you’ve been doing so far. This time, though, we won’t be giving you the answers.

We’ll send you the assignment via e-mail later this afternoon. (SHOW ASSIGNMENT) It will look something like this. Instructions on how to submit your responses are in the assignment. Make sure you e-mail your responses to us no later than 11:59 PM EDT, on March 26, 2017.

Once we get your responses, we’ll look them over and give you some brief feedback, and send you instructions on how to claim your CE credit.

# Questions?

Again, thanks for sticking with us, and I hope we’ll see you again at one of our future classes.

It looks like we still have some time left, so I’d be happy to take a few questions right now, as well.