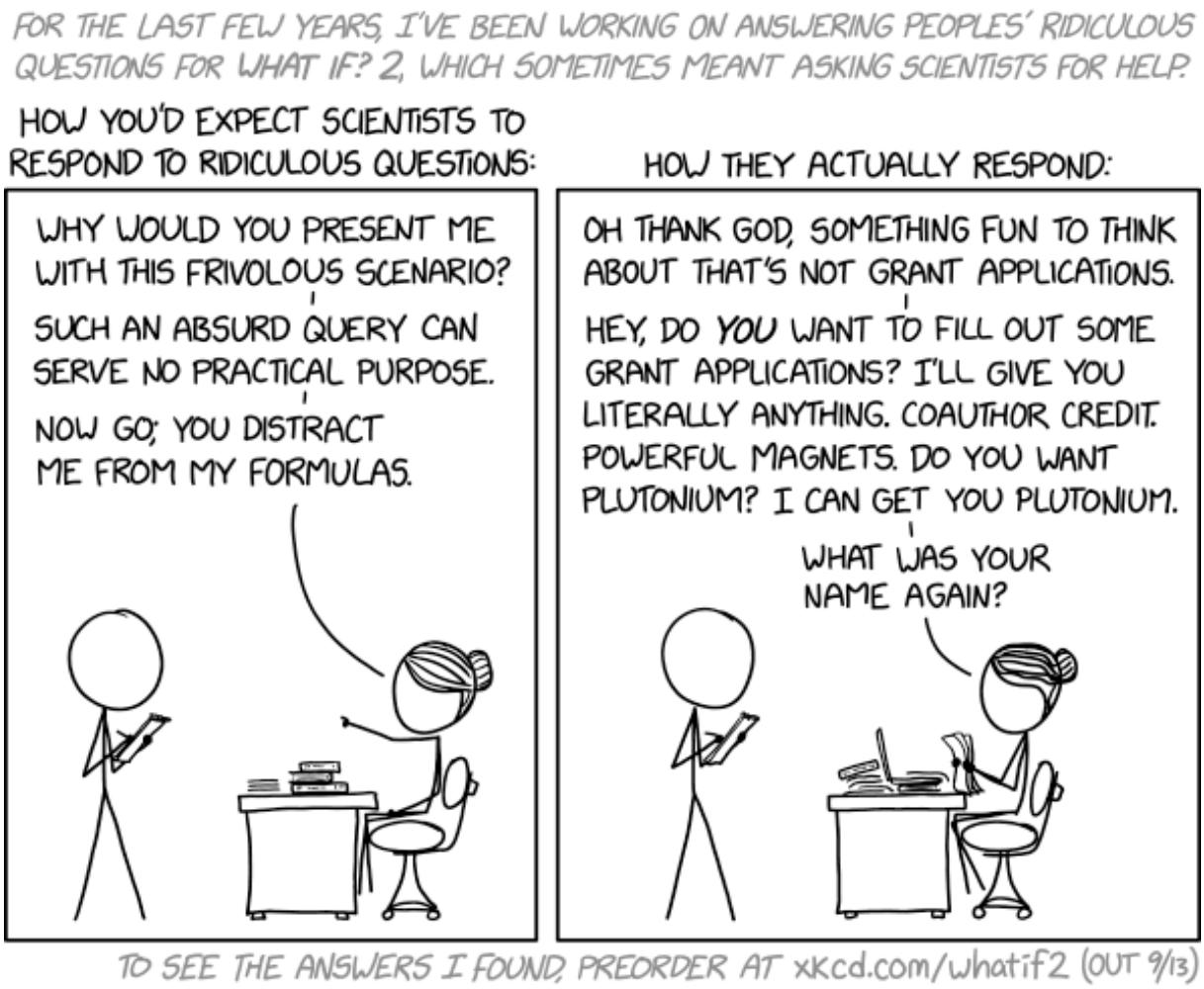
Volume 01

		4				9		
			8				2	
3	8		5					7
6							9	2
		8		1		6		
9	7							4
4					6		1	9
	3				5			
		1				3		

FUNNY PAGES


FOR THE LAST FEW YEARS, I'VE BEEN WORKING ON ANSWERING PEOPLES' RIDICULOUS QUESTIONS FOR WHAT IF? 2, WHICH SOMETIMES MEANT ASKING SCIENTISTS FOR HELP.
HOW YOU'D EXPECT SCIENTISTS TO RESPOND TO RIDICULOUS QUESTIONS:
WHY WOULD YOU PRESENT ME WITH THIS FRIVOLOUS SCENARIO?
SUCH AN ABSURD QUERY CAN SERVE NO PRACTICAL PURPOSE.
NOW GO; YOU DISTRACT ME FROM MY FORMULAS.
HOW THEY ACTUALLY RESPOND:
OH THANK GOD, SOMETHING FUN TO THINK ABOUT THAT'S NOT GRANT APPLICATIONS.
HEY, DO YOU WANT TO FILL OUT SOME GRANT APPLICATIONS? I'LL GIVE YOU LITERALLY ANYTHING. COAUTHOR CREDIT. POWERFUL MAGNETS. DO YOU WANT PLUTONIUM? I CAN GET YOU PLUTONIUM.
WHAT WAS YOUR NAME AGAIN?
TO SEE THE ANSWERS I FOUND, PREORDER AT xkcd.com/whatif2 (OUT 9/13)

Passive Aggressive Work Emails With My Toddler About Dinner

Hi Goof!

Just circling back on whether we're set for the 6pm with pesto noodles.

Thanks!

Mommy

Ahh sorry – just seeing this. Since it's so close to the mtg should probably go ahead and reset.

Apologies. Have a great rest of your night.

Goof

Actually, all good if you're still avail!

Pesto is ready and waiting for you whenever. Want me to carry you to the chair or are you good to walk?

M

Hey, Mommy

In all transparency, is there kale in the pesto?

Let me know.

G

Hi G!

Can definitely check for you, but in the meantime, for the sake of expediency are you good to get started?

M

I'll stand by while you confirm.

Hey, Goof

Ok. Good news and bad news: The good news is I'm hearing the pesto is the best yet. People are very excited about it (including Dad, not to name drop) and I'm so glad the stars aligned and we could get you and noodles in a room together. On the kale front, it's looking like a yes. Regardless, from a big picture standpoint re: growth/digestion/etc., we all think it's definitely the right move strategically.

We'll go ahead and confirm you for eating the pesto for 6:15 since you've got a hard out at 6:30 for bath.

Hey!

Totally hear you. I think sadly after dealing with some personal stuff on my end it's just not going to work out and I hate to do this but possibly won't make the 6:30 either. Send my apologies to bath!

Best,

Goof

Hi Goof,

So sorry to hear about the personal conflict! I hope all is okay! We'll be thinking of you. I'm hearing that the meeting is sadly not flexible. And unfortunately neither is bath because it's going to be booked at 6:45 for (again – not usually this name droppy!) the baby.

One development: I got word that we can do one episode of Bluey on the iPad during the 6:15 if that changes things, but (sorry – they were really set on this point) you have to be eating to watch it.

Thanks,
Mom

Oh - That does change things. Two episodes poss?

G

Great!! Sadly it’s only one because of the bath double-booking later.

Apologies!

Mommy

K

-Goof

Amazing! Have a great rest of your night!!

Same to you.

I’ll see you at 3am.

See you then!!!!

The Grudge Report is a reader-supported publication. To receive new posts and support my work, consider becoming a free or paid subscriber.

TECHNOLOGY

What are the ways you go about getting comfortable with a new codebase?

> Tools like Eclipse's Java Browsing Perspective Visual Studio's Object Browser make it easier to browse the codebase by components.
>
> What are the ways you do it? Is there anything similar available for functional programming?

muzani wrote:

Yup, I've stopped looking for the entry point in SPAs and apps. Often the navigation flow can be extremely convuluted.

Android introduced navigation graphs [1], which were meant to solve this problem. But what happened in reality were instead of arrows point to each screens, there would be multiple islands that were teleported in from random places.

davidatbu wrote:

Interesting. Thanks!

buro9 wrote:

It makes even more sense for web apps. It will tell you how it came to be that a web page received the request, what context it has, how the cookie is resolved to an auth context, how to access any query args, what is available to support the page.

cornel_io wrote:

Good resource! Alas, the specifics really depend on the language.

jarusll wrote:

So indexing all the components and finding out all the interactions between them. This is exactly what class browsers do, they index all the classes and messages. The interactions could be described using an example/documentation.

This reminds me of Pharo which does all of the above, indexes classes, messages and has a rich documentation support.

jarusll wrote:

This is very well written. 1-3 describes my experience of tinkering in Pharo except I haven't actually built anything with it.

ahurmazda wrote:

If you are lucky (working with a mature codebase), tests are my number one go to when getting started. I need to know input/output of things. I work in ml space so ymmv. This allows me to make small changes and check my assumptions as I gain more confidence around the codebase

smugma wrote:

We have a tool that was built by a third party. They did an excellent job but for various reasons we needed to change vendors. I didn't hire the old or new dev teams, so it wasn't my role to tell them how to come up to speed. Early on they said they wanted to redo all the

test cases, which seemed off to me (it's too abstract, and why redo test cases for parts of the code that are unlikely to be modified). I said something but didn't push it.

Someone on my team has been giving the dev team demos of the functionality and thinking behind the product a few days a week. My one request at the beginning was that they should learn enough about the product to be able to give a demo back to us. It took them about 2-3 weeks (maybe 8 45 min overview sessions from my team, which owns the product requirements), but it showed that they know what it is the tool is supposed to do.

They spent another 3 weeks "getting comfortable" (6 weeks from start) they finally felt comfortable to start implementing small features and bug fixes. I'd have preferred that they start fixing bugs right away (it might take 2-3 weeks to fix the first bug because they need to figure out how to get access to systems, documentation, deployment, etc.) because it's more tangible, but I know I'm impatient and let them do it this way. It seems to work ok so far but will be another month or so before I can decide whether or not they are actually competent. I guess the good news for them is they (team of 10 in Eastern Europe) aren't being bugged by the client, and if they actually are good, should be enjoying the freedom to do things their way and implement their own processes.

smugma wrote:

How do you think of #1 in the context of a web app? Each page is essentially a different entry point into the code.

I guess the landing page or authentication page could be considered the equivalent, but I'm not sure those would hit your goals to understand flow of data, etc. ?

TrackerLounge wrote:

Background: I have worked for several companies with systems that have been activity used and developed on for 30+ years. The databases have been complex, containing 1000s of unique tables and views. These databases often have hundreds of in-database user defined functions that can be 1000s of lines long. The code bases have been in multiple languages, using multiple technology stacks, and multiple git repositories. These

projects have been a mix of monolithic and micro-service architectures. Oftentimes, there have been multiple front-ends (to meet different types of users' needs – e.g. internal company users, vs external public users). The user applications have typically had ~100-200 pages to do specific types of actions. It is common for the UI pages to have to comply with laws, government regulations that have changed over time. These laws and regulations comprise of 100s-1000s of pages of text. It is common for a single button on a UI page to do very complex actions (e.g. batch jobs) , including building out large data-structures, touching (e.g selecting, inserting, updating, deleting) 10-50+ database tables. It is common for UI pages to only be used at certain parts of the year or once every few years. This makes it much harder to "know about" those parts of the system. It is very common for UI Button actions to have to "know" and enforce details found withing the laws and government regulations. Aside from the application code, we also have large ETL (extract, transform, load) processes to build up data warehouses. There have also typically been a large number of batch jobs to do regular data pulls and updates from other systems.

Given this background, "What are the ways you go about getting comfortable with a new codebase?" Here is some of what I feel/think/do:

1. In these companies, it is "understood" that a new hire will take about a year+ before they know enough to be useful. These systems took peoples entire careers' to learn about and bill. It is "understood" that a new hire will not learn the system "overnight".

2. Even knowing that it is "impossible" to know the system quickly, I have a personal desire to contribute to the company (e.g. be of use) as quickly as possible. It is very important to have reasonable personal expectations. Chill out and allow yourself time to learn. You can't avoid paying the TIME cost, to learning a complex system. You can't skip the Time/Exposure cost required to learn a complex system. If you do skip the TIME/Exposure cost, you are fooling yourself into thinking you "know" something you do not. It will show.

3. Get access. You need access to the code base repository(s). Get access to the database(s). Get usernames

and passwords (if allowed in your company). Get login credentials to your applications. Get access to the documentation store(s), ticket tracking system(s), applicable laws and regulation(s) locations, etc.

4. Get a development environment running. Checkout the code. Get it running on your local machine. I want to be able to run in debug mode where possible.

5. Define what it means for you to feel like you know the system. For me I feel like I "know" the system when: a. all the names and acronyms are familiar (every company and system has their own "language" b. I can look a random page in the system and known what database tables are referenced when the page loads, and what database table(s) are changed when the UI buttons on the page are pressed. c. I have a mental model of the data and business validation that will occur when a given UI button is pressed d. I am comfortable enough in the language(s) and technology stack(s) to make changes, review code, and deploy code e. This standard is pretty easy for me to "say" but takes YEARS of hard work to achieve.

6. Commit to keeping a log of your daily activities. Writing what you are doing, seeing and learning can act as a form of team coding with yourself. As you write, you are forcing yourself to "teach" and "explain" what you are doing and why. This activity helps me internalize the system quicker. This can help a lot if you are in a company that requires you change problem contexts a lot during the day or week. Also take LOTS of screen shots in your log (using Snagit or ShareX, etc).

7. Commit to creating system documentation. This can be in markdown, confluence, word, etc. It can be in a git repository or file system. It must support screenshots and page links to related pages. I tend to do UML diagrams in Drawio or Visio (exported as images) but if your tool supports UML all the better. I like to organize my system documentation by UI Page. For every major UI Page, I have a system documentation page of the same name. It may have many child pages (depending on how complex the page is). This simple structure means that I can quickly find the documentation that I have 6 months later when working on the same page again. Having a documentation space to put documentation makes it easier for me

to create more documentation.

8. Depending on the company, you may ask to job shadow another developer for a period of time. This can mean spending an hour a day or 8 hours a day with them in a team coding environment. Ask another developer to do this. If they say yes, you can learn a lot more quickly about the system.

9. Depending on your company you may have leeway in how you use your time. You may be given issue tickets on day one or you may be sheltered from the storm of requests and given time to learn for a month to a year+.

10. If you are given leeway to research and learn (which is rare), one of the best ways for me to learn is to try to build a mimic system. I look at the existing system and try to build my own system to do what it does. If I can produce a screen that loads data that looks like the original system than I have a high degree of confidence that I "know" how it works. If I can produce a button that saves data (and does all of the complex validation, database changes, etc) that the original system's save button does, then I have a much higher degree of confidence that I "know" how it works. As I mimic the system, I attempt to document what I am learning about the original system. These notes can become invaluable later (when in a time crunch to fix some critical issue). Building a system mimic makes it much harder to "fool yourself" into thinking you know what is going on. Let the compiler and screen comparison be a mentor. They can be brutally honest but effective teachers.

11. If you are not given such leeway to research and learn (which is common – people pay you to produce), you can learn a lot be reviewing prior completed issue tracking tickets. You can see the most common topics and rough patches in the system (for that time period in the year – problem hotspots change over the year(s)). The existing system documentation (if any exists) may be very helpful. If there are Unit Tests, these can be very valuable learning aids as well. As you work issue tracking tickets you will be forced to learn details about that specific area of the code. It may take longer to get a "wholistic" view of the system this way (as opposed to building a mimic system) but it will eventually get you to a similar level of understanding, via a

different route.

12. Some of these company's architectures have and support Unit Tests. Count yourself luck. Make use of them and add to them. Other company's architectures do not (or make it really hard).

13. Spend as much time as you can playing with the Application UI. Pretend to be an end-user, loading pages, entering data, clicking thru the process and saving. You will see a lot of validation errors. You can learn a lot about the workflow and pain points in the workflow. You can learn a lot about how the front end effects the database. Systems tend to get better if you (as a developer) "eat your own dogfood" that you make your end users eat. Be in and use the application(s) on a daily basis. This will help you learn the company language and help you communicate with your users. It will also help you see why users are having trouble with a process and perhaps see ways to make the workflow better. It will also expose bugs and give you an opportunity to fix those bugs and learn more about that particular part of the system.

14. One of the most useful things I can do to learn a new system is to build "UNDO" features. I am not certain why this isn't more 'popular'. It feels like it grants SUPER POWERS to me. If I am looking at an Application UI button that does a complex Batch process, the best way to "know" that I know what it does is to build a script (and another button) to undo what that Batch button did. Many of the systems I have worked with do not have an UNDO button for complex processes. This means that everyone is scared to run them because they do so much. This means they do not get run or tested as often as they should. This builds up additional fear of clicking the button. If I can create an UNDO script, I will have to document everything it does. I have to "Know" every database record that it changes. Once I have the UNDO script, it becomes easy to run the button to do the complex batch process and it becomes easy to get back to a virgin state again. The easier it is to run, the more roundtrip testing you can do and the more likely it is for you to make the code better. This all can greatly reduce fears of that process in the future and can greatly speed up supporting the feature in the future.

I am sure there are could be more added to my list. Of course every company and every position is different. You'll have to figure out what makes sense in your situation.

Hopefully it gives you some ideas. Good luck learning your system(s).

rmk wrote:

Here are a few techniques that I use, in addition to requesting talks from people who know the code base, asking for code walkthroughs if they know the code, and just generally learning the terminology for various things in the domain (i.e., getting familiar with the Nouns and Verbs of the system):

- Check out the source code and look at how the codebase is laid out. i.e., the directory names. This often gives me an understanding of what is vendor code or third-party dependencies, test, and 'core' source code.

- How to build the product, and what artifacts are produced. This will tell you a lot about where to start looking.

- Take a small bug where further information is requested to help triage/solve the bug. This could be reproducing the issue by clicking around or performing a workflow by issuing commands, or rooting through logs and database entries to ascertain the state of the system. This is often an instructive exercise because it requires that you understand how the system is deployed, which is always a good thing to know, and how to troubleshoot an area that you know little about, which you will often be called upon to do. Reading the logs gives you a pretty good idea of the system initialization sequence, particularly in a cloud product. Ditto with looking at the core metrics or example traces of the system. Traces, if available, instantly lay bare a 30000 ft view of the system before you.

- For web apps, the router file. There is usually a file that contains various route definitions and the entrypoints to them. This is a great start for figuring out what links to what. Something basic like a simple GET of a collection or a health check is a great way to get your feet wet.

- For web apps and others that use a database, the database schema. Often I just do a COUNT(*) of tables and look at the schema tables that contain the largest number of entries.

- Unit Tests. For a particular functional area, these are an excellent aid to

understanding what the expectations are and how they are tested. I also write unit tests for areas that do not have them as a way to get familiar with the codebase.

- Results of Smoke Test and Integration Test runs. These often give you a bigger picture idea of what the system does in relation to others that surround it, and the major 'compartments' of the system as it were.

- Fixing bugs of varying complexity. This is an excellent way to instantly get familiar with building, testing, code reviewing, and putting through standard precommit testing your change. The change itself is incidental; the things that you learn during this process will help you get productive quickly.

- Writing small features that are very self-contained and interact with one or two areas of the system. This helps you understand a few system areas inside-out, and you can slowly grow your understanding of the system by doing features that touch newer areas that you'd like to know.

funwie wrote:

In addition to already mentioned steps,

0) Read the code base docs (or README).

1) Pair with someone with knowledge of the code base or ask them to walk you through the code base.

2) Identify the public interface to interact with the app/api. How do consumers use the software. Play around with the app or api to get a sense of how things link up.

3) Identify various tools used in the code base(db, messaging, external api, etc). Now you know each tool is setup somewhere and used in one or more places.

4) Identify the patterns and conventions used (CQRS, mediator, dependency injection, middleware, pipelines, logging, etc). Now map the flow of each public interface using this knowledge.

tedmiston wrote:

These are great.

A 4th way I would add is: If you need to make a minor change or understand how one specific function is expected to work, search for its unit tests and start there.

mikewarot wrote:

Old school - print it out, use a pencil to make notes, and a highlighter as needed.

dominotw wrote:

just start working on features/bugfixes and learn as you need and only the things you need. You'll be really slow in first couple of features/bugs but will get faster as you go. Set this expectation with your manager.

bravetraveler wrote:

I tend to use the thing, find a behavior I don't like, then build familiarity pulling at that specific thread

xupybd wrote:

Learn the domain. Learn as much as you can about the business logic, the problems the code solves. It's so much easier to do this outside of the code. I've spent days trying to fix bugs only to find the business logic was wrong.

sys_64738 wrote:

Read the design docs.

mstipetic wrote:

If I start with these tools I get a feeling I understand how things fit together, but when I want to add something new I realize my understanding is pretty shallow. I force myself to manually open files where things are so I get an understanding of the conventions and principles in the project. By far my most valuable tool is grep (it's super fast in vim) and I grep the code to see where certain functions and such are used. I use the lsp tools later when I'm used to the project.

FILM

Blissfully Gazing: An Encounter with Sayombhu Mukdeeprom

> “It’s all about feeling,”

Thai cinematographer Sayombhu Mukdeeprom told me several times whenever

I tried to frame the nuances of his methodology with conceptual notions.

His words, however filled with ambiguity and elusiveness, might in fact seem to be the key to describe the general premise of the films he has worked on as a cameraman—it is, indeed, all about the feelings, participation, and intuition. After all, the work of Apichatpong Weerasethakul, Luca Guadagnino, Miguel Gomes, whose films Sayombhu has shot, revolve around a certain reciprocity—the images link with the tactile, offering a space for the audience to immerse themselves in the images: just as real as imagined. A meditative gaze floats with Sayombhu's camera in carefully designed master shots, following the characters in a tender rhythm, accompanying them from a safe distance, inviting participation. This can be said particularly about Apichatpong's films, with whom Sayombhu worked on a majority of his projects, sharing the creative process for a universe of shape-shifting narratives, starting from *Blissfully Yours* (2002)—although some sources invalidly suggest that their first collaboration was a docu-fiction *Mysterious Object at Noon* (2000), which was debunked by Sayombhu himself during our talk—and framing it with Colombia-set *Memoria* (2021).

Recently awarded with the International Film Festival Rotterdam's Robby Müller Award for outstanding achievements in cinematography, Sayombhu has helped create the layered visual worlds of Guadagnino's *Call Me By Your Name* (2017) and *Suspiria* (2018), as well as Miguel Gomes' *Arabian Nights* trilogy (2015). From Apichatpong's envisioning of Thailand, to Guadagnino's Italy and Gomes' Portugal, Sayombhu finds himself well in different environments—that includes different types of projects, from arthouse and expanded cinema to impressionistic shorts and commercial features. The latter, in particular, made him reflect on the agency of independent filmmakers in Thailand and the general lack of funds, which resulted in establishing a post-production studio in Bangkok called White Light (founded with the editor and fellow collaborator, Lee Chatametikool), which focuses on supporting upcoming indie projects from the region of Southeast Asia.

Sayombhu's work is known for his preference for film over digital, spontaneity over hard-planning, and medium and wide shots through which the Thai cinematographer enacts an observational stance that becomes an inherent component of Apichatpong's narratives. Sayombhu's sense of visual style derives from the willingness to interact with the environment on set; there's always a margin so that the things may go with their natural flow. This finds its salient translation in a choice of lighting—mostly coming from natural sources, oftentimes hiding the characters behind the shadows, in glimpses, in worlds in which there exists something dim and unexpressed. There's an act of performance in it, as well—to cherish the desired angle of the light might as well mean the inevitable: to wait for the sun to come up.

Sayombhu offers his audiences cinematic images filled with simplicity and comfort. His slow-paced and rhythmic shots maintain a steady flow—filming from a dolly is another of Sayombhu's landmarks—thus becoming a canvas for the haptic particles to be immersed. The image connects with the sound, the sound with the memory, and so it goes; the results are blissful poetry, but then again, it also achieves a snapshot of a realist rawness. About all these components that build into a refined body of work, I got to talk

with Sayombhu Mukdeeprom over a Zoom call. Connecting from Boston, where he was preparing for the next Guadagnino project, he started our sessions by speaking about Boston's light, admitting with a sense of disappointment that there isn't a lot of it there.

NOTEBOOK: There's this lovely picture in the Fireflies book about Apichatpong Weerasethakul's *Memoria* where you are sleeping on the shooting set.

SAYOMBHU MUKDEEPROM: Oh, the famous picture—let me tell you about this. It was taken by my assistant. The thing is when we were shooting *Memoria* in Colombia, the set was quite relaxed. We had to wait for the sun to arrive and it was around lunchtime. I didn't feel hungry at all, so I said to my team to go and have a bite and I would wait at the set and stay with the camera. We were in the middle of the city, so we just couldn't leave the equipment there. And I wanted to be there, that's it; I wanted to be near my cameras. And when they came back after several minutes, they saw me there, but resting in a state of half-sleep.

NOTEBOOK: You said on some other occasion that when you don't shoot the film, you sleep a lot. And since Apichatpong's films deal with the notions of sleeping and dreaming a lot, I was wondering if you dream about your work? Do the images come to you in your sleep?

SAYOMBHU: These are not dreams, but a constant circle of images that keep running through my head. But I don't perceive it as a good thing. I talked with a doctor once who told me that most cinematographers suffer from high blood pressure. The work is stressful, you know, and it makes the images run in my head. When I'm asleep, the process is still ongoing, it keeps spinning around with the images. Sometimes even if I wake up, I have to gather all the ideas—it's like reaching out for the lost snapshots of the last night. But some of the ideas are good. They come to me in my sleep, so they derive from somewhere organic, natural. It's somewhat similar to the state of meditation. There's no scientific theory behind that, but for me, it's a part of my working process.

NOTEBOOK: Speaking of meditation—does it help you organize the space in

your head before the shooting?

SAYOMBHU: I think it helped me a lot in general because overall it helps to deal with the stressful parts of work. To me, it's just a tool to keep myself calm. It also helps me realize what is stressful, what stays inside. In a way, you have to protect all of that and deal with it later. And if you keep on working and working, then there's never time to go through it. You just keep on going.

NOTEBOOK: You started with Apichatpong but you also worked with other arthouse filmmakers from Europe, Luca Guadagnino and Miguel Gomes. How did these collaborations start?

SAYOMBHU: It was very simple. I just got contacted by them and that's it. Nothing special. They reached out to me and I talked with them and I liked them. That's how it started. Very simple. With Gomes, it was that he had some problem with his cameraman and he was looking for a new one. And because someone recommended me to him, he called me and that's it. It's normal for the film society—we intertwine.

NOTEBOOK: During Rotterdam's masterclass, many people you've worked with over the years congratulated your award, oftentimes saying you're like a part of the family. There's this evident feeling of togetherness that comes out of those words, appreciating the fact of mutual trust and collective effort. How does the aspect of closeness determine your perspective on the sets?

SAYOMBHU: My point of view is that making films is demanding. That makes people the important key in the process of making them. If you are comfortable with someone, then the work will stop being just the thing that you have to do. It will become a pleasure. It's like with touch—if you're comfortable with someone touching you in a friendly way, then it's an entirely different level of comfort you have there. That said, I always intend to work with people whom I trust because there's a lot of thinking about other people's thinking. And you don't want that, you'd rather rely on your trust. Trust is part of being family. Apichatpong is family. If he calls me, and if I'm available, I don't ask him about what he's about to do. I know that whatever that is, I want to be there for him. I trust him and I feel him. And if I don't know someone, then we talk, get to know each other's feelings. That's vital.

NOTEBOOK: The beginning of your career started in Thailand but then you moved to different countries: Italy, Portugal, the United States. Do the language and environment influence the way you work?

SAYOMBHU: Actually, no. And I'm lucky to say this because the visual is a universal language. The same can be said about the technical parts of it. Same camera, same gauge, same lenses. Typical things. I communicate in English to talk with my team about our work. And truth to be told, I don't need to understand the words as the image speaks for itself.

NOTEBOOK: Your work consists of mostly fiction feature films but the *Arabian Nights* trilogy by Miguel Gomes is a project that leans towards a documentary lens. What is the difference for you in terms of navigating between fiction and documentary?

SAYOMBHU: Documentaries are more fluid. This is crucial for everybody to understand that. With fiction there is a responsibility to carry out the work; as a cinematographer, I need to deliver something that is required of me. That's the basic difference. When I work on a documentary project, we can rely on everything that is given to us by natural circumstances—the sun, the noise, anything organic—and that's perfectly fine. But with fiction films, especially big projects, my job is to cover things up, deal with the obstacles, and create the image on top of it.

NOTEBOOK: I was wondering if you could talk about the process of finding the locations. That's the part of the cinematographer's work that is a bit behind the scenes and it seems it's a very important aspect of the methodological approach for you and Apichatpong.

SAYOMBHU: I think it's just how Apichatpong wants to work. Most of the time he gives me a call to go with him and check the spot. But if he doesn't want me there, he will go all by himself or with other people. But he always contacts me first about it. And when we're there—it's all about the feeling; it's me sensing his thoughts and feelings about the film. I go with Apichatpong to absorb the environment and the locations. And it's often the case that we do it without the script even written. This is when we start talking about the concepts

and ideas he has in his mind. It's often that it starts with the location, it connects with some idea. Then we come back with the script and start thinking about the place in frames. Since *Blissfully Yours*, there were two Apichatpong's films that I couldn't shoot because of my other projects: *Tropical Malady* (2004) and *Cemetery of Splendor* (2015). But even then, I was prepared to go with Apichatpong for location scouting. He sent me the script and we would still go together to get the feeling of these places.

NOTEBOOK: One of those peculiar locations that reappear in Apichatpong's films are institutions, mostly hospitals. It's a very difficult place to shoot the film in. A hostile environment that attacks with difficult lighting; one that overwhelms with sterility and whiteness. But at the same time, as in the case of *Syndromes and a Century*, the whole film is set in a hospital and it's rendered in an amazing way. How do you approach the visual (in)capability of the hospitals?

SAYOMBHU: I don't see myself as someone who approaches shooting at the hospitals in a different way than any other cameraman. I think it's always about having control over lighting. It's like you have to set a mindset of thinking in terms of lighting range. When there are scenes to be shot in a hospital, you might want to shoot them during the afternoon because light cools down a bit. This way you can control the whiteness of the interiors and turn off the fluorescent lighting which is organically bad for the camera. So it becomes a process of finding the right tone; you change the lighting back and forth until you find the right contrast. I see this as an art of dealing with your location; it's all about making it more interesting, more appealing, and more fitting to the demands of the bigger picture.

NOTEBOOK: The hospitals are way different than in most of the films. The setting has this peculiar energy, somewhat of magic in it.

SAYOMBHU: I'd say it's because Thailand is in a tropical climate, so that's not magic. [*Laughs*.] I mean, I grew up in a hospital, similarly to Apichatpong, perhaps even more because my father was a rural doctor changing from one place to another, usually from a small facility to even

a smaller one. Apichatpong's father worked for a big hospital. And I guess this energy might come from the fact that we don't perceive hospitals as sad places. Yes, people come and go, and a lot of them die there, but this is the first and fundamental thinking of Buddhism. We are born. We get old. We die. It's a circular motion. It's our understanding of the reality we live in. Another thing is my memories. As I said, I grew up in the hospitals, so the snapshots of the hospitals from our films are heavily influenced by either mine or Apichatpong's images from the past. For instance, that recurring sequence in *Syndromes and a Century*, where they go through that bridge-like pathway connecting two branches of the hospital—that's precisely taken straight from my memory. It's all there for you to see!

NOTEBOOK: The amount of research done for *Memoria*, available in Fireflies book devoted to the film, is outstanding. The interdisciplinary nature of it, the details—you're emphasizing the improvisational side of working with Apichatpong, but there was so much prep work done beforehand. How much indeed is there space for feelings and going with the flow?

SAYOMBHU: It's all mixed. For me it's all about looking into the eyes: Apichatpong looking into mine, asking in his head, "You like it or not?" Simple as that. Sometimes I would just say, "Oh, Apichatpong, that's impossible!" But then we talk. There's a lot of going back and forth in our work—we're always giving ourselves this margin to try things out. And we talk a lot—with each other and the rest of the crew. There's always room for a "yes," even if something seems distant and impossible. The whole process seems to me like cooking. If you're good enough, you won't need the recipe at some point. You'll just cook and admit it's pretty damn good. Or too salty. Or it will make you want to throw it away.

NOTEBOOK: Akritchalerm Kalayanamitr, a sound designer for Apichatpong's films, also told me that he considers himself a cook. Coming back to the places—in Colombia, you shot mostly in Bogota and Pijao. Particularly the latter seems to be a place with very distinctive energy—very slow-paced, with lots of green integrated with the urban fabric and many older people

inhabiting the city. Did you have any specific approach to how you wanted to convey the city landscape?

SAYOMBHU: It always starts from the script. We have the story that we have to tell. Then from that, there's the process of constant searching. I think when I was there, I never stopped thinking of the city. I kept looking and exploring for something that might be good for the frame we wanted to capture. Again, it was based on hit-or-miss. We would find some place only to let it go in the end. There were also many moments when we could trust our feeling. Sometimes the scene might turn out to be perfectly synchronized with a certain place. Who knows?

NOTEBOOK: How do you get to know the place?

SAYOMBHU: I think it's important to get to know its rhythm in the first place.

NOTEBOOK: Do you limit yourself to observing or do you talk with people as well?

SAYOMBHU: Mostly observe, because I think my part is not about people. It's about the nature of the environment. And this is something I love to do; it's my favorite part of preparation work, the most exciting one.

NOTEBOOK: Do you take photos of the places you explore?

SAYOMBHU: Very rarely. I do take photos, but mostly in my head. It's because I'm preoccupied with thinking about the technique. Once I touch the camera, I can't think of anything but the technicalities, so I have to put it away to think clearly about the potential of a place.

NOTEBOOK: Since you shot many scenes in the natural environment, I was wondering how much you had to adjust the shooting schedule to Colombia's unpredictable weather?

SAYOMBHU: In general, this is about the capability of a cameraman. The light is your resource and you have to know how to rely on that. To operate around your camera you have to know how it works, right? The same goes for the sun. It's like possessing the technical knowledge of the sun, like getting a manual on it that is explained through patience. You have to prepare for it. If not, then you can always resort to creating everything artificially. But it always depends on the project itself or the kind of set you work on. *Memoria* demanded we get that knowledge.

And when you don't have the light, you have to create it and deal with the artificial one. When I first read the script for *Uncle Boonmee Who Can Recall His Past Lives* (2010) I noticed that in the scene when they enter the cave for the first time, Apichatpong wrote something like, "The cave is so dark so we see nothing." I thought, "Oh, good, fine! I can easily do just that!" [*Laughs*.] But even with that, we had to do something about it; we had to make the audience see that darkness.

NOTEBOOK: *Uncle Boonmee*'s night scenes look beautiful—there's this uncanny grainy touch to it. You're known for opting for film over digital. Why's that?

SAYOMBHU: For me, it's actually all about the camera. It's connected with the rawness of the image I want to look at because when you're looking at the image through the optical viewfinder, it becomes so different from the picture appearing on the tiny video monitor in the loop. My biggest concern is to look right into the optical, as there's nothing that filters your reality there. Through the lens and optical path, the image makes all this way to your eye. With cameras, there are lots of processes that happen all at once before the image arrives at you. They make tricks to your senses, too. So when I look through the optical, I get that sort of feeling that strengthens my interaction with actors. Because of the *photogénique* aspect, actors trick the gaze of the camera as well. They use different angles, adjust their movements, pause, and change the way they look or disappear. And it's up to you what you do with your camera, how you frame them. You can go dancing with the image, play with it a bit.

NOTEBOOK: Do you recognize lots of mistakes from the past when you look back at your old work?

SAYOMBHU: Of course, every day! I'm always thinking that I could do this or that. This is because I learn every day. But it's come and go. I think of it as a part of Buddhism, like a circle of things.

NOTEBOOK: Do you think that Buddhism had any influence on your sense of aesthetics?

SAYOMBHU: Yes, I think so. A lot and on many layers. It's mainly because my craft is to deal with the present time.

Not the past, not the future. I'm right here, right now. You do have to prepare for the scene, but what's more important is when you capture the moment. It's all in the moment! When you're filming you cannot stick to the past. You have to let go of everything you had preconceived in your head and focus on the present. Here and now.

NOTEBOOK: Is there any peculiar and distinctive image of Thailand for you? When I asked Akritchalerm about the sound of Thailand, he responded with "crickets and car traffic."

SAYOMBHU: Oh yes, the sound is definitely correct. As for the image—I think it would be mid-noon sunlight. That's exactly what I love about Thailand's light. There's no such lighting elsewhere than in Thailand. Like in *Blissfully Yours*, you have that particular angle of the sunlight that casts a shadow on the characters' faces. Nobody likes that, but for me—I love it! It's so difficult to get that angle in Europe. Almost impossible, because the sun's angle is way lower than in Thailand.

NOTEBOOK: Your first project was *Blissfully Yours* which initiated your long collaborative relationship with Apichatpong. What has changed for you during all these years?

SAYOMBHU: There are of course lots of things that have changed. First of all, I changed as a person, I think. I'm getting older, right? I perceive the world in different categories because of my experience. The same goes with Apichatpong and our films. But in terms of the spirit of making the films with him—I don't think anything has changed. I feel that we just keep a young spirit, you know, we're still exploring, experimenting.

PHILOSOPHY

I. The Marriage of Concept and Vibe

One of the core teachings of mystical traditions is that good and bad are interdependent, sometimes to the point of being "flesh of one flesh." Evil isn't the *privation* of good as classical Christian theologians, following Plato, argue, but part and parcel of it. William Blake captures the sentiment when he writes of the *marriage* of heaven and hell. Neil Gaiman and Terry Pratchett popularized this notion in *Good Omens*, which recently got adapted into a TV series.

In Kabbalistic thought, the *other side*, or *sitra d'achra,* refers to the demonic dimension of the good. Carl Jung articulates a similar idea when he posits how all human strengths come from and also lead to a *shadow side*, a point of weakness. We find Jungian archetypes all over media: In Star Wars, Luke Skywalker and Darth Vader share an intimacy, despite being on opposite sides of the Force. Harry Potter's powers derive from the primal wound inflicted upon him by Voldemort. The concept is also operative in vaccination, whereby a small dosage of a disease is the very thing that protects against it.

The ethical, legal, and political implications of mystical thought are often bothersome to normies, who think that the fungibility of good and evil gives license to moral relativism, or worse: the elevation of evil to a moral ideal. This is the theme of Gershom Scholem's studies on antinomian thinkers like Sabbatai Sevi and Jacob Frank, who believed they could bring "Redemption through sin."

But you don't have to look to fringe examples to see the point. The Talmud makes it banal: without an evil inclination, it says, we wouldn't build houses or reproduce. Some basic sense of egotism, of the desire to conquer, some basic amount of lust, is needed to get the ball of life rolling.

It's not so much that desire is neutral, but that it is both positive and negative at the same time. The French psychoanalysts call the phenomenon of painful pleasure *jouissance*.

II. The Marriage of Concepts and Vibes

Once you see good and evil as interconnected, a lot of one-sided debates look different. This is part of what Hegel teaches through his method of dialectic, a high-minded approach that itself become the basis for Bill Clinton's and Tony Blair's "Third Way" politics in the '90s.

One of the great polarities that I see—and that doesn't break neatly along a left/right political axis—is between concepts and vibes.

 Zohar Atkins@ZoharAtkins

Kant said it best: Concepts without vibes are empty, vibes without concepts are dumb.

August 2nd 2022

13 Likes

Ok, actually Kant used a different word: *intuitions*. But *vibes* is more contemporary. Both intuitions and vibes appeal to emotion, to immediacy, to something we know without reason or justification.

How does a pizza master know when the pizza is ready? By following the recipe? No. By vibing. The master craftsman is not bounded by the concept, but the vibe, of pizza. Yet vibes also lead astray. They are mushy, often prejudicial. And because they are harder to put into words or arguments, also harder to scale.

A good heuristic: vibes can be learned through apprenticeship, concepts can be learned from books.

Vibes are charismatic. Concepts are rigorous. A vibe without a concept is snake oil. A concept without a vibe is an empty suit.

When Kant was writing about concepts and intuitions he was writing about human consciousness, about how the mind constructs meaningful experiences out of the chaos of sensory data. But his teaching could also apply to culture.

A world in which people are technocrats only would lack vibe. A world in which people just follow their emotion could lead to all kinds of terrible places, from the wide embrace of conspiracy theories to outright fascism. Political rallies are a vibe. And as Burke knew, the task of politicians is

to provide theatrical entertainment. But the wellbeing of a society can't live on cheer and outrage alone. Policies should be evaluated on their merits, not on the basis of pre-determined ideological, tribal allegiance (vibe).

III. The Marriage of Populism and Elitism

Why am I translating Kant's intuition as vibe? In part, because I am trying to express stylistically a marriage of high and low culture. This marriage is not only tactical for me, but a sincere expression of something I believe: we need to bring elitism and populism together, without compromising on either. We need to elevate everyday life by giving conceptual analysis and shape to it; we also need to uncover the beating heart that abstract thought, at its worst, denies or represses.

Elitism tracks with top-down authority, populism with bottom-up. We need both. Vibes are bottom-up. Concepts are top-down. Concepts alone detach us. Vibes alone have no staying power.

Some great artists manage to combine both high and low. Shakespeare is an obvious example—his plays borrow from classical tradition yet also contain plenty of gallows humor. A typical Roberto Bolaño story describes a drug dealer or prostitutes who reads Pindar and Alcaeus.

Democracy is basically the enshrinement of low culture as an ideal, while aristocracy is the enshrinement of high culture as an ideal.

Today, we need the combination more than ever—we need a culture that takes the responsibility of elites seriously, and one that doesn't take them too seriously. We need a culture that finds dignity and wisdom in the people, but also doesn't let the people have the last word on what is good, true, and beautiful (for these things cannot simply be a popularity contest).

In short, mysticism needn't be fringe or antinomian. Mysticism, in the Hegelian variety I've articulated (Hegel was influenced by Christian Kabbalah) is practical. It's about bringing opposites together, about

finding points of collaboration where other see opposing teams.

Whether we are talking about consciousness as in Kant, God, as in Kabbalah, archetypes as in Jung, art as in Shakespeare, or politics as in Hegel, we need to find the hell in heaven and the heaven in hell. We need to vibe with concepts and conceptualize the vibe.

MONCHILLA

This is an urgent hot dog public service announcement: You need to bánh mì your next dog. Immediately. Forget the mustard and the relish, and prep some pickled daikon and carrots instead. This is, legitimately, the best way to eat a hot dog. After I was done rubbing my belly from happiness, I was actually pretty angry that I didn't try it earlier and because I care about you, I want to save you from the same self-irritation. A Bánh mì-ed dog is officially my favorite way to eat a hot dog and I'll never turn back.

If you haven't had the pleasure of devouring bánh mì, allow me to introduce you. Bánh mì is a wonderfully balanced, flavor-packed, Vietnamese sandwich. Different versions abound but the chief ingredients are consistent. It's usually made with a marinated meat (often grilled pork or chicken), sliced cucumber, pickled daikon radish and carrots, sprigs of cilantro, a smear of pâté, and mayo (or other sauce), all securely nestled into a split baguette. What you end up experiencing is each of the five pillars of flavor in one sandwich–umami, from the marinated meat; salty, from the sauce; sweet and sour, from the pickled daikon and carrots; and bitter, from the cilantro. While pâté is traditional, it's not on every iteration of bánh mì, and if it's not suited to your palate, it can be omitted. I like to top my sandwich with sliced jalapeño but many folks opt for a squirt of sriracha.

The match-up makes perfect sense. Hot

dogs are salty, juicy, hand-held, meat sticks that can provide a similar umami balance to the marinated meat in bánh mì. I know the classic hot dog toppings are beloved, but (now, be honest), the summer wiener can get a tad repetitive. Maybe you're a ketchup girl and you branched-out and tried pickle relish, or you always do mustard but last week you tried cheese. While these explorations are good first steps, it's time to spread your wings and bánh mì.

A traditional bánh mì uses a baguette for the bread, but if that's not your vibe you can absolutely stick with the softer packaged hot dog bun. I tried both and, for once, I didn't have an obvious preference. I thought the pickled veggies would make the hot dog bun soggy, but that was not the case. Of course, things will get messy if you're silly about it and pour on the pickling juice, but I fished out the pickles with fork, gave a slight shake to remove excess liquid, and dropped it on my dog with no problems. The baguette version was equally as scrumptious. The chewiness of the bread made me slow down a little and bask in the wonderful flavors of the bánh mì dog, and the bread was a little bit taller so I could fit more jalapeños and cucumbers in the bun.

This recipe comes together in about 15 minutes with mostly easy-to-find, staple ingredients. (Note that the picture shows the amount of ingredients I used for prepping two hot dogs, but the following recipe is for four.) Daikon radish is available in asian grocery stores and most other grocery stores with a decent-sized produce section. They add a light flavor to the pickled vegetables that I love, but if you can't locate them, just go ahead with the carrots.

Recipe

Ingredients:

- 4 hot dogs
- ¼ cup julienned raw carrot
- ¼ cup julienned raw daikon radish
- 1 teaspoon of coarse sea salt (¾ for fine salt)
- 2 teaspoons of sugar
- ¼ cup rice vinegar (use white distilled vinegar for a stronger bite)
- ¼ cup warm water
- 4 segments of baguette (cut to hot dog length) or standard hot dog buns
- Mayo (to taste)
- 4 cilantro (full stems the length of the hot dog)
- 4 long slices of cucumber
- Fresh or pickled jalapeño slices or sriracha (optional)

Heat the hot dogs in any manner of preference (maybe even a dirty water dog). While they're cooking, prepare the pickled root vegetables. Put the julienned carrot and daikon in a bowl or jar, add the salt, sugar, vinegar, and water. Mix slightly to help dissolve salt and sugar. Pickle for 15 minutes. During that time, stir occasionally to rotate the carrots and daikon, ensuring even pickling.

Slice the baguette lengthwise, but don't go all the way through. Assemble the sandwich by first spreading a bit of mayo into the baguette. Add the hot dog, tuck sliced cucumber onto one side, and pickled veggies into the other side and on top. Add a sprig of cilantro and jalapeños or sriracha, if using. Enjoy immediately. Never look back.

Building on a vision more than 20 years in the making

Amigos, hello!

You know, I have a lot of restaurants…30! Crazy. If you ask me to choose a favorite of course I can't…but Zaytinya, which I opened nearly 20 years ago in Washington DC, was one of the first and so it has a special place in my heart. The restaurant is an expression of my love for the amazing foods and flavors of the Mediterranean…Turkey, Greece, and Lebanon. And now, two decades later, just last week we opened the doors to Zaytinya at the new The Ritz-Carlton New York, NoMad in New York City.

The restaurant is also special to me because of Aglaia Kremezi, who is a brilliant Greek food writer, cook, and journalist…the Julia Child of Greek food. When years ago I was beginning to think about Zaytinya, I took a trip to Kea, the beautiful island in the Aegean Sea where she lives. I wanted to meet her and to learn from her so that the recipes we served were authentic. We had never met before, but we connected in an instant. We spent weeks together cooking where I wrote down everything she taught me, and took many pictures to bring back with me.

She taught me many things…to make Dolmades (stuffed grape leaves, a recipe we still use today!) and to always *always* roll phyllo by hand and never rely on the packaged brands…and so much more.

Aglaia has been our muse and the one who helped guide us and develop the Greek side of our personality…we call her our Greek Grandmother. To this day, she is involved and keeps us grounded in tradition. Zaytinya's Concept Chef Michael Costa, who has been with us for 11 years, still travels to her cooking school to work on recipes with her, and we have a regular call with her to go over menus and talk about food and what she's cooking.

This type of culinary exploration and collaboration is a huge part of how we

create food in our restaurants. It is not just me, José, doing the menus. It is all of us. The one thing I always emphasize in my kitchens is that no chef exists in a vacuum. Every kitchen is a village of cooks, and we celebrate the contributions of everyone. Our restaurants bring people with different backgrounds and skills together…that collaborative culture is the foundation of our food.

For Zaytinya, in New York City, our "village" was lead by Chef Costa along with New York Zaytinya Chef José Ayala, pastry chef Abby Naguit, ThinkFoodGroup Executive Pastry Chef, and Andres Lara, the Research and Development Pastry Chef for ThinkFoodGroup, and cooks like Selim Topal, who is from Turkey. We traveled, we read books, and had a lot of dialogue with Aglaia, to be as authentic and true to the stories and the regions as we could.

Our Zaytinya village!

On the menu you will find many new items, in particular for breakfast, a meal we don't serve at Zaytinya DC. I love how the countries in the Mediterranean bring in so many flavors to the first meal of the day—chilis and tomatoes and salty cheeses, it's amazing. Take the **Shakshouka** we make…a braised tomato and chicken stew with poached eggs, onions, and sweet peppers. Over the top we have pickled chilis, preserved lemon, parsley, and lots of olive oil. Incredible!

Menemen, which originates from a small town in the Izmir province, is a very traditional Turkish scrambled egg dish with tomatoes, spring onions, green peppers, star anise, Marash

pepper, and feta, with some chives and olive oil on top. And have you tried **Çilbir** (pronounced jil-bur)? This is a dish that dates back to the Ottoman sultans, as far back as the 15th century: poached eggs on toasted pita with whipped yogurt, chives, and a Harissa chili crisp. Delicious.

The Çilbir at Zaytinya with poached eggs and Harissa chili crisp.

Let me also tell you about our desserts! Other than one dish, the Greek Yogurt with Apricot, which we have had on the menu since day one, all of the desserts are new and are like nothing we have ever served before. **Islak Kek**, which translates to "wet cake" is a recipe you'll find dogeared and stained with cocoa powder in the kitchens of Turkish grandmothers. It's a milk-soaked chocolate cake, sort of like the Turkish version of Tres Leches, that is super light and fluffy, topped with barberry compote, labneh, and rose espuma, with a red fruit and floral-infused granita and a finishing touch of sumac. Boom!

Our Kunefe, a dish from the South-eastern parts of Turkey that border the Mediterranean Sea, was something we worked on for a long time, trying to make it as gooey as possible. Selim Topal, who is from Turkey and is a cook at Zaytinya, helped us develop this recipe, which we finally feel like we have perfected… it has crispy kataifi (fine pastry strands of crispy phyllo dough), the gooey cheese, and pistachio marzipan drizzled with a rose and lemon scented syrup. It is finished with a salted lemon yogurt and crushed pistachios.

The Greek Yogurt Parfait with Apricot has been on the menu since day one.

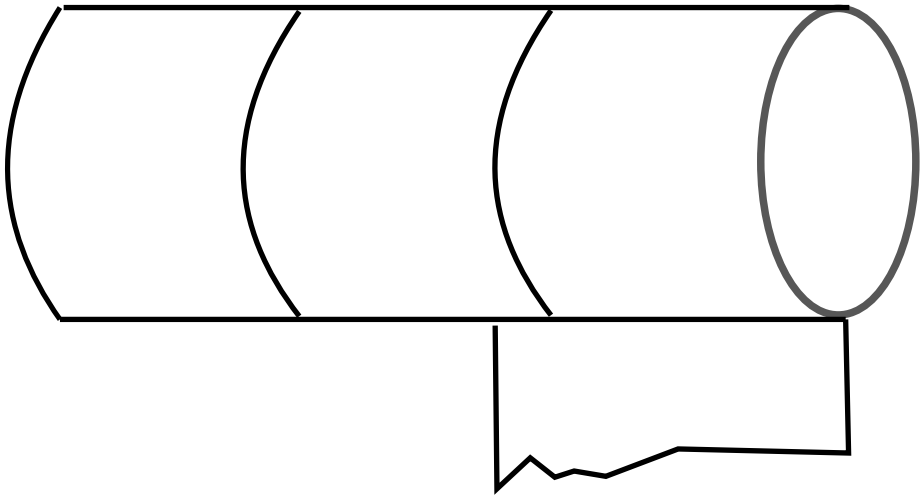
Galaktoboureko is a traditional Greek phyllo-wrapped semolina custard that's baked and soaked in syrup. We turned it into a citrus summer dessert with lemon semolina custard and gave it some salinity with dehydrated black olives, and added crushed pistachios and lemon sorbet.

Traditional **Kaymak** is a clotted cream from Turkey that is heated slowly so the cream rises to the top and slowly thickens over a long process…18-24 hours! The result is a very thick cream, like a pudding, but when you eat it it's very light like a cloud. For our version, which Selim also worked on with us, we paired the Kaymak, which we source in Brooklyn, with a saffron-soaked pine nut meringue cake.

Amigos, I hope that you are feeling hungry now and ready to dig into some dishes from the Mediterranean. For paid subscribers, we will have two amazing classic Zaytinya recipes for you on Wednesday…a summertime pomegranate cocktail and the classic red pepper and feta spread, htipiti, that everyone loves.

"De las mejores ideas [...]"

"jajajajaja bring a towell jajajajajajaj 4242424242424242 jajajajaja inside joke only jajjajaaj IQ > 170 para entender la referencia jajajajjaaj Muskie déjame chuparte la polla porfaaa daddy muskie mmmmmm"



"I mean lets be honest here. It most likely is not gonna be a worldwide success but i can def see it get popular with the hipster/ get back to monke types"

¨Está tuanis el librillo, osea si lo viera en una librería y valiera menos de 3 euros de fijo lo compraría¨

5.80€

ISBN 978-0-4356-5678-5