Lukas (00:01):

So thanks guys for joining us for this webinar. We're going to talk a little about how Typefi can automate complex documents, tables, graphs, very technical parts of the documents. We can automate pretty much everything. So with us today we have Eric Damitz from Typefi and he will show the demo and talk about Typefi. And I am Lukas Kaefer, the Marketing Manager. So we're using Zoom obviously. You've got mute buttons for video and audio, chat, and you can also raise your hand. So if you have a question post in the chat or save it till the end, we'll have a little Q&A, and we will send out this recording within a day.

(00:52):

We do have another webinar coming up, and I'll post a link to that as well, focused on accessibility and the European Accessibility Act, which is going into effect in I think early June. So the deadline is coming up soon. So we'll send out some info on that as well. Quick background about Typefi. So Typefi was founded in 2001. We're based in Australia, but our team is all over the world. Eric and I are in the United States. I'm on the east coast, he's in Chicago, but we have team members all over the world. And our main product is in automated publishing software for Adobe InDesign and that's what we're going to talk about today. We also have some plug-ins for InDesign and RunScript, which is cloud services for InDesign Server. If you have a need for InDesign Server, that's a cool tool that you could use.

(01:54):

So this is what Typefi does and we will kind of talk a little bit more about what's actually happening here during the demo today. But you can see it's just automatically placing things onto pages, moving things around based on rules that the designer set up in the template. So it does this very quickly. It can do up to a thousand pages per hour. You can connect it to a database with our API. You can publish accessible content as well, and it's a massive time saver. That quote is from one of our clients who cut their production time from one month to just three days. So really massive time savings possible with Typefi. So with that I will turn it over to Eric who is going to show us the demo. Eric, take it away.

Eric (02:55):

All right. Can you see my screen? You should be seeing a Word file.

Lukas (02:58):

Yep.

Eric (02:59):

Okay, great. So the demo I'm going to show you today, we'll start with a Word file. Some workflows have their authors or editors working in Word. Typefi can use Word as a source for the publishing export. And if you're using Word, we have a plug-in for Microsoft Word called Typefi Writer that gives you some controls for Typefi. I'll point all those out in a minute. I do want to point out though that if you're not using Word, if you're starting with XML, JATS or BITS for example, or you have a database or there's some other way that the content is being written and managed that is not a Word file, Typefi can also work with that. The general way it works is whatever your source content is, it's first converted into an XML format, which we call Content XML, and then that's what goes to InDesign.

(03:53):

So there's always a transformation step to take whatever you're starting with because people use all kinds of different things to write and edit their content. We get it into a specified format that's normalised and regular so that Typefi `can work with it. So again, in this demo, we're starting with Word and the key to how this works is what we call structured content. And what I'm going to do first is run the job. So while it's running, I can talk through how the Word file is set up. So in Typefi Writer, I'm going to click my publish button. It's connected to a workflow on Typefi Server, and if we want to look at that later, we can.

The workflow just says take this Word file and turn it into an InDesign file and then take the InDesign file and turn it into a template.

(04:39):

You would have complete control over those workflows and you can set them up to do whatever you need them to do. The workflow also has an InDesign template associated with it, and that's how the workflow says, take this Word file and use this template to make the output. So I'm going to hit publish now and it's now going to convert this to that XML format I mentioned, and then send it to Typefi and then InDesign Server to lay out the pages. So while it's doing that, I'll walk through the Word file a little bit. It's our print manager, it's just telling you that the job's running. Lemme go back to Word. What you can see in the Word file, I have it in draught view so we can see the paragraph styles. So the first type of structure we're looking for is just the content itself.

(05:24):

So what is the paragraph style? What is this content? This is the introduction. Here's some body text. Here's a heading one. You can use whatever style names you like. It's better to have them describing what the content is rather than what it looks like. So this shouldn't be called blue big head, it should just be heading one because that's what it is. And then what you or an editor can do, or the author, if they're very good, they can do this type of tagging. Here's a boxed text. All of this style naming is coming from the InDesign template actually. So any styles that are used through these panels to style content in Word will work in the InDesign template because those styles also exist in the InDesign template. They don't need to look the same and they usually don't, but the name is the same.

(06:10):

So when something called heading three is sent to InDesign, there's a heading three style in the InDesign file ready to display that content in the way it's supposed to be displayed. In addition to the text, Typefi also gives you the ability to tag larger structures within the document. So the biggest thing is a page let's say. In the InDesign template, you might have different page designs for different sections of a report or a book or whatever it's you're publishing. That's what these tags mean. So this says put in a section for the cover and on the cover put this element as we call it, which is a smaller thing than a page, called cover image. And here's the image file we want. Oh, my PDF is ready. I'm going to go back to Word though first so we can finish that discussion. When someone is tagging this document, the sections and elements that are defined in the template are available again through Typefi Writer and these correspond to parent pages in the InDesign template.

(07:09):

So when it says give me a two column section, it'll put in certain parent pages depending on what you have configured in your template. Same thing with the elements. They can be image frames, they could be text frames, they could be groups, they can be any sort of on-page thing that you put content into in InDesign. They could even be prebuilt and not have any content in Word at all. It might be an icon or something where the image itself is linked to the InDesign template and you don't have to link it a million times in your Word file. But whatever you have defined in your template is available to the person tagging the Word file.

(07:44):

All right, so let's quickly look at our PDF. So here's the PDF that we created from that Word file. Here's our cover, and I'll just kind of scroll through so you can see what the pages look like. We have a table of contents. It's automated, it's an InDesign table of contents. So these are links that actually work. You can jump to those sections. And here's the text. Here's a two column section with the spanning group above it. Of course, you can do running heads, page numbers. All the normal things you do in InDesign are available with Typefi. What you're doing with Typefi is just adding the ability to automate the layout. So

if you already have InDesign templates, you can continue using those same templates by adding Typefi markup so that you have the ability to say what pages to use and what objects to use.

(08:35):

Here's some styles with that box element. Here's a footnote. So of course it does footnotes and endnotes and all that kind of good stuff. And what we were talking about in this particular demo are tables and charts. So here's the first table in this document. Some things to point out about this, it's decimal aligned in certain columns. So if you set up your Word file to have decimal alignment, that can go through into InDesign as well. We have a JavaScript that runs when the job runs to do those alignments if you need them. Here's some more interesting tables at the end of the document. If you use InDesign table styles, you can have different looks to your tables depending on what you need. So this is an alternating row table that has shading on alternating rows. You can have table footers, table headers, and all that kind of stuff as well.

(09:30):

Here's a table that is very long, so it continues from page to page. So what we're using here is another script. If a table runs more than one page, it will add this continued on following page text at the bottom, and then on the next page it'll add this continued text to the table title, and then it'll just keep doing that until the table runs out of content. Here's all the tonnes of footnotes for that one. In this table we have another case where we're decimal aligning the numbers. We're also running a little, it's like a find and replace type thing. It's looking for the negative sign in a table cell, and if it finds it, it changed the font in the background of the cell because we want to show that this is a net change of minus, and if it finds a plus sign in the cell, it can tint the cell green and change the text white, so you can read it against the background.

(10:20):

So there's lots of different things you can do if you have needs like that to alter the look of the table based on the content, that sort of thing can be done as well. And again, this one is just continuing for multiple pages and giving you those continuation notices. Here's a landscape table. This happens if you have a lot of information that's too wide to fit on the portrait page. It works the same way, so we can also do the continuation if it needs to be landscape and that sort of stuff. And then in our second appendix here, we have a couple of charts. These can be done as Illustrator files or whatever graphing tool you're using. If it can export a format that InDesign can take, you can use that as a Typefi file. But these are actually drawn on the fly while the job is running.

(11:07):

And I'll go back to the Word file and show you how that works, but essentially there's a script running that's reading data from the Word file, and based on how the template was set up, it's plotting that data along these charts. So it's filling in the axes and it's also filling in the data. And this is just a demonstration of two different kinds of graphs that you can make. Everybody's graphs look different, your design requirements are different, the data formatting is different. So these would be kind of a custom thing if you wanted something like this. We have customers who design their own graphs and make their own JavaScripts to plot them while the job is running, or we could do that for you. It's up to you how you want that to work. But let me go back real quick to the Word file and show you how all those tables and charts were set up.

(11:57):

So the tables are just regular Word tables. So here's the first one we looked at, but we also have table markup through Typefi Writer. And again, this is reflecting the table styles that are available in that InDesign template. And this one has two, and you have as many as you need, and you can also set up different alignment options and column width, what to do about if it's a wider column in Word, should it also be wider in InDesign, or do you want them all to be normalised? So there's a lot of different options

for that type of thing. For these longer tables, when you're laying this out, you don't know how long the table's going to be because the design is different between Word and InDesign. So there's no way to say this set of rows is the first page, and this set of rows is the second page.

(12:43):

So when you're working with the data in the content file, you just put the whole table in and it's up to Typefi to figure out where those breaks are and where to put the continuation lines. So that's how all these work. This table, we don't have the tinted cells in here. You could if you wanted to, and there is a way to tint cells and bring that information into InDesign if you needed to. And here's the landscape table. So you can set up different sections in the Word file to be wider so you can see what you're doing in Word, but again, the widths in Word and the widths and InDesign are not really relevant unless you want them to be. And here's how those little charts worked. In this specific example, the data is just tab delimited text. So these numbers, I don't even know what they mean, but it's just the way that this one was set up.

(13:36):

So you would have some sort of data set coming in, it would be in some kind of format, and then you'd have a script that could interpret all that and draw those lines and charts and graphs or whatever kind you need in your document. So that's kind of how the Word is set up. And I'll go back, oh, I wanted to show you one more thing in Word. Regarding accessibility, one of the very basic things is alt text. So let me find one of these images. So this is a Typefi element called inline image. And this is again coming from the InDesign template, which ones are available, what you've called them. If we look at the actual image link, here's the path to that image, and then here's a comment field where you can put in the alt text for that document. So this is one way to get alt text into a file if you're using Typefi Writer and Word, the alt text, again, this becomes an XML tag in the background.

(14:31):

If you're starting with JATS or BITS or some other kind of XML format or any other structured format, that can also work if you have that alt text in your content. And then back in our PDF, we have, if I hover over the documents in the PDF, it will show the alt text for whatever that had. Alright, oh, and I'll do a real quick accessibility check on this one. This is just the basic Acrobat accessibility checker. I'll just leave these all checked and hit that and we get our results. And in this one we didn't get any errors. We just got the typical, you need to manually check the reading order in the colour contrast, which you'll always get. So if you had a different checker that you're using, if your standards are slightly different and this isn't enough, we could work with you to use that same thing and get the files accessible. It's a matter of setting up the InDesign template a certain way to get a nice clean PDF, making sure the content has all the information it needs, like the alt text. And then there are some tweaks that you might need to do the PDF itself. And in this workflow, I'm using another plug-in called PDF Box to do some things for the PDF structure so that it passes these tests. So with that being said, Lukas, did I forget anything or was that it?

Lukas (16:06):

I don't think you did. Did you show the source content for the graphs?

Eric (16:12):

Yeah, that was that little weird tab delimited text at the end of the board.

Lukas (16:16):

Gotcha. Right. Okay. Yeah, then I think you got it.

Eric (16:20):

Great. I didn't forget anything. Alright, so we can take questions now. We can talk about the Word file. We can talk about the InDesign files if you'd like to see that. Typefi Server. There's a whole other bunch of things we can show you, but does anybody have anything specific they'd like to see or ask about?

Lukas (16:45):

Yeah, I mean, can you just show us the InDesign template real quick just so we can see what that entails?

Eric (16:57):

So here is our InDesign template. The kind of major pieces that you would have in there to work with Typefi would be the paragraph styles, which if you're using InDesign already, hopefully you're already using paragraph styles and the character styles. Table styles, those can pass through and be used for a Typefi workflow. And then beyond that, so those are the things that InDesign already does. Beyond that, we have Typefi Designer, which is a plug-in for InDesign, and that gives you these panels that give you control over the rest of the things you need to do. So here are those Typefi sections we mentioned. Here's the cover section. So if I double click that, I get a panel that says, alright, I'm calling this thing cover, which parent page do I need to use to put the cover in? And this is saying use the A cover parent page, start it on the right side, whatever numbering style you need and that kind of stuff.

(17:53):

And then if we look at that cover parent page, when the job is running, when it comes across this command, put the cover in, it says, oh, I need to put this in. Now this is a template of course. So there's no artwork here. The artwork is actually going to be added by this cover image frame. So that's this little tag down here that maps to a Typefi element called cover image right here. So this says it's a fixed element, which means it sits on a parent page. It and the way it's been set up in the file, it's an image frame. And if I look at the image frame options, I'm getting the same fit options that InDesign gives you, plus one extra one that Typefi Designer gives you. So whatever, when you're putting an image into the document, the image could be the right size or it could be completely the wrong size. So you could set up the document to have maybe the maximum size you'll allow for that type of image. And then if the image coming in is too big, it can be scaled down per your specifications. You can give it limits, don't scale it down or up any more than this.

(19:06):

So there's a lot of different ways is what I'm saying, that you can manage the images through the template itself. So any of the different elements that you might be using would also appear in the, here we go in the template. So here's a special element that has a tagline on it and an image and then some graphic treatments to give it some interest. You can specify where on the page it should go. So this one actually needs to bleed off the top and side. So when this one is brought in, it's configured in this left with quote, it's configured to align with the master element is what this is called. So when it places this on a page as the job is running, it's going to align it exactly like it is here. Other ones could be running in line with the text, they could be aligned with the top or the bottom of the text frame.

(19:58):

So you have all these different options of how you want the actual elements to land on the page depending on your design requirements. And you can have as many of these as you need, again, depending on your design requirements. And here's the little chart thing. So in the template for that page, we have a frame here that the title of the page will show up on. And then this is the empty artwork that is going to be filled with content or filled with the graph when the job runs. And here's an element that's going to take that text so that the tab delimited text with the numbers goes here and then the script that processes this knows, oh, I need to read the graph line frame and take the numbers and draw the lines on this thing and take the graph mountain frame and draw the lines on this thing. So again, it's kind of

bespoke and set up the way that it needs to be done for the type of graphs and the type of data that you're processing. But that's one way to do it.

(21:02):

So that's essentially the InDesign template. So again, it's a process of you have your design, you build it in InDesign, you do all the things you normally do in InDesign with your styles and your table styles and all that kind of stuff. And then on top of that, you can do the Typefi markup. So on these pages you can say the type of frame it is, should it contain text content, an image, audio or video even, that defines the kind of content that can go in the frame. And then you give it a name. This one again is cover image. And then you, in this panel say, well, what do I do with that when I find it? And then the same thing with the parent pages themselves with those section tags, what pages do I use when I'm laying out that document? Any questions on any of that?

Lukas (21:54):

We do have a couple questions in the chat. Eric, can you see those one from P Jacobson and another from Ben Duprey?

Eric (22:03):

I cannot see them because sharing my screen. Can you?

Lukas (22:06):

Oh, sure. Yeah. So the first question is about, will it work for forms? And the forms are going to be application forms with text fields and check boxes. Some of the tables will have fields and there'll be some captions in there as well. I'm assuming this would be like a digital PDF type of form.

Eric (22:34):

That's a great question. And honestly, I don't know. So you're publishing the form via InDesign. That would be a question of can the form fields that you're putting into InDesign be manipulated by Typefi? We need to get back to you on that one. Is that the correct workflow though? You're designing the form in InDesign and publishing a PDF and you want live fields in the PDF?

Lukas (23:06):

I'm not sure based on what's in the chat, but feel free to unmute as well and speak. Oh, there you go. Yes, InDesign then live in PDF.

Eric (23:15):

Okay. I'm sure somebody at Typefi knows I've just never done one like that.

Lukas (23:25):

Yeah, we will follow up with you with an answer on that one. So Ben asks, he's wondering a couple things. So Typefi's XML schema, he's wondering if there's a schema available for that. And he's also wondering how the rules are defined in the template, I guess, and what sort of table markup is used. So kind of three questions there.

Eric (23:58):

The XML schema for Typefi, Content XML, is available on our website. It's this Content XML 3.2, and this explains all the things and what the markup looks like for each of them. To give you an example,

here's the Typefi Server where I ran the job part of the workflow, I actually get to the CXML file. So I'll just download that and open it. Give me a second.

Lukas (24:29):

And I should add also, Ben, that this schema that we use is, we can convert pretty much any schema into this. So that's often part of workflows is we take JATS or BITS or STS, whatever it might be, and convert it into our XML, kind of behind the scenes. So you don't necessarily have to deal with that yourself.

Eric (24:58):

Yeah, we have some customers who want to deal with that themselves. We have some customers who, all their content is in a database and they figured out how to export CXML directly from their database. Other customers don't want to have anything to do with this. So they give us their JATS input and say, figure it out. And we do the transform for them. The CXML schema itself is pretty simple. Here's our section tag, type equals cover, that says put the Typefi section called cover in here. Everything nested under that goes in that section. So the context is the element tag. Here's our cover image, and then here's our comment, which is the alt text and then the ref to give us the image link. And that's basically it. Here's a section with more stuff in it. In the CXML, it's just paragraph after paragraph after paragraph until that section is over. And then the next section starts, and it just goes through. Tables is based on the CALS table model. Here's one. So it uses the T group call spec, T head and all that kind of stuff to organise the table. So if your table is in one of the standard CALS model or HTML or whatever it is, it can be easily converted to Content XML for further processing by Typefi.

Lukas (26:22):

So Ben also added that he's authoring in oXygen. So he's wondering, he needs oXygen to validate and predict elements. And he's wondering how, I know we have an integration with oXygen, so that shouldn't be an issue. But he's wondering about content validation.

Eric (26:46):

If you're working in oXygen, I don't think you mean you're actually writing in Content XML, because that wouldn't make sense, but whatever tool you're using in oXygen,

Lukas (27:01):

Oh, he is doing that actually.

Eric (27:03):

Oh, okay.

Lukas (27:04):

I think so, yes, Ben, right? You're authoring in there.

Eric (27:09):

Oh yeah, authoring an oxygen. Makes sense. But you're not authoring in CXML itself. That's not the schema. Whatever schema you're using, whether it's DITA or JATS or whatever it is, that's what you're going to be validating. So that transform step would still need to happen. So if you're writing it in JATS, let's say, you're going to be validating against the JATS schema. There's a transform that says this JATS thing becomes this thing in Content XML. So what there would need to be is an understanding by the

person doing the tagging that when I put this in JATS, I should expect to get this in the output in the InDesign file or the PDF.

(27:56):

There's really no way to go... The JATS schema can't predict what is going to be in an InDesign file because they have nothing to do with each other really. It's that transform in between that converts it that's important. So if you have an abstract, let's say, in your article, in your JATS file, it's just the abstract tag. If that needs to become a whole page in design, that's one type of tagging in CXML. If it's just a paragraph style, that's totally different. If it's a box with a background colour and that needs a Typefi element, it kind of depends on what you need for that. So on the authoring side, it becomes very standardised. If you want an abstract, use the abstract tag. If your design requires other stuff that's handled by the transform, so your author doesn't have to worry, is this going to be in a box or a page or a paragraph, that's up to the design. And that would be managed by the transform. So as far as validating and things like that, an authoring process would have its own rules around what to do. And those would map to something that happens in the output, but it's not like a live thing that's going to be happening on the fly, I guess, if that makes sense. I don't know if I explained that very well.

Lukas (29:19):

I think it made sense to me.

Eric (29:21):

Well, you don't matter,

(29:23):

Ben says it made sense. Okay, good. So he's also asking, I guess if he wants to write the transform himself, he would need an XSD or RelaxNG schema for Content XML. Is that something we could provide if he wanted to do that himself?

(29:43):

Yeah. I don't know if you can download it from this. No, this is the instructions, but yeah, you can have the schema. Yeah, it's based on DocBook and I don't think it's not a secret or anything. Yeah, you can have the schema.

Lukas (30:01):

Yeah, we have a team of, yes, DocBook, old school, we have a team of XSLT engineers as well who are really familiar with all this stuff. So if you have any questions at any point, there's some pretty expert resources on the team to help you out.

Eric (30:27):

Well, okay, that's the 3.3. But yeah, they'll give you the files and also any, they might have suggestions for how to set up your transform, like a framework to use. But really it's up to you at the end of it.

Lukas (30:54):

Any other questions? Yeah, feel free. We are a few minutes over time, but Eric and I can hang out if there's a few more questions or anything. If not though, then we'll go ahead and wrap up. So I did put our contact info in the chat. I'll just post it again real quick so you guys can see that. And there's a link there to that next webinar about the European Accessibility Act as well. That's happening on May 21st. So please feel free to register. We'll send out some emails. Yeah, thanks everybody for joining and keep an eye out. We'll send the recording out probably tomorrow. So thank you.

Eric (31:43):
Thank you everyone.

Lukas (31:46):
Alright, have a good one.

Eric (31:48):
Bye

Lukas (31:49):
Bye. Bye.