Speaker 1

So today's day five of our development on Creature platform guided learning and we continue to work with some parts of Creature user interface slightly moving forward to so to study some server side items. And also I need to tell you about Classic UI and possibly you'll be interested in migration from Classic UI to the Freedom ui. So what I need to tell you first today, this is about details. Because this is a very important piece of Creature configuration and you will face it almost everywhere. Let me show you what detail is go to customer 360 because it's a good example of details. We go to accounts section and then we can open any account record. Detail is separate entity which keeps some data in connection to our main record. Here we can see an example. Main record is account and detail is account address.

Speaker 1

And you can see that for one main account record we can save unlimited number of addresses. So we can put unlimited number of addresses here and they will be usually displayed in a list. And the reason why Creature has details is need to save additional data in connection to our main entity. So our main entity is a single record and then we need to keep information for example about different types of addresses for our customer or different emails. Or maybe it's just simple list of contacts connected to this account. So you should understand that detail is usually based on a separate entity. So account is one entity, Account address is a separate entity, it means separate object and it means separate physical date table. Usually we have one too many relationship between details and main records.

Speaker 1

So one account connected to as many detail records. And this physical connection is provided by use of lookup columns, correctly filled in for detail records. Vikrat, thank you very much for confirmation. Thank you for working with your redis and I'm really happy that you finally made your system working correctly. Thank you. And so details, they are usually connected with main entity with the help of lookups, lookup columns. And each detail record has information about its parent account. And that's how system can filter through all addresses and show only addresses related to this currently selected data record. Details are very common in creation. If you will turn on bigger product like sales service and marketing, you will see details almost everywhere. And we have big entities like contacts or accounts and they have plenty of details like addresses in accounts.

Speaker 1

We have contacts list, we have also emails or sorry, we have probably it's better to say communications. If you go to contacts open any record, you will see communications detail. So here you see we have mobile phones, email, business phone, but in general we have plenty of Supported communication options and communications are saved as a detail connected with main entity in context. We also have detail named job experience. And when we have several entities, sometimes they are connected with each other with the help of lookups. And that's why and that's how we can make some details. For example, we can look at accounts and see list of connected contacts because contact has a reference to account. Let me show you here Account select. And now you see that we have contacts and they have reference to account.

Speaker 1

And that's how we can select a company and to see all its connected contacts. So usually connection is one too many. Maybe some of you will ask about is it possible to make a connection one to one or data connection many to many. Physically this is possible, but you should use for one to one you still will have to use separate object. And for many to many we will have to use supporting additional object which will use two lookups for two connected entities. So the easiest to implement and I think the most commonly used case is use of one to many relationship. And now let's return back. Let's return back to our reality section. So we will study and we will learn how to make

details.

Speaker 1

And some of you who already participated in our develop and in our customization course, you already know how to do this and you already saw building details. So this is nothing special for you. But some of you who never attended any of creatio training sessions, so you need to know what detail is how to build it because it's almost the same important as knowledge how to build sections. So I will show you step by step example of building details and I will explain why we need it. The main reason is our business requirement to keep a lot of connected information for our main record. And this connected information usually should be saved in a separate entities. So when we talk about details, we talk about separate connected entities. So let's do some quick and simple example.

Speaker 1

In our realty business, we definitely need to speak with potential customers. We need to provide some kind of show for them or our apartments or houses set for sale or rental. And we need to organize these actions. So we need to organize such visits, such conversations. And it's better not to intersect and overlap different visits at same time. So we need to have some kind of schedule for future visits connected to a certain realty record. So it looks like we will have a separate entity called realty visits. And for each house or apartment we will have a list of planned visits for future we can do it with a separate object and I recommend you to create new details for the first make an object for it.

Speaker 1

So let's go to our configuration section select our package where we will work on maybe we focus on objects first and we will create manually create new object. So new object requires new good working friendly code. We can call it something like realty visit title will be also good. Sorry realty visit inheritance it makes sense to inherit only from base entity in order to get just six standard system columns here keyfield historical columns. Okay, we do it inheriting from base entity and we can create our own custom columns for our new entity. One of the first column will be connection with parent realty because our visit will have no sense if we are not connecting it to any reality. So let's create new lookup column.

Speaker 1

We can call it like parent reality we can name it parent realty so title to be lookup based on reality from freedom UI USR realty and we can make it column required because it's not good idea to keep it empty if we create any valid visit so this colon will be required. We have no default values here, but we have a special lookup behavior. Normal behavior means when we delete value from corresponding reference table from realty Having such connection with any other data table will prevent our delete and we will have a foreign key reference error message. Because we can't delete reality if it has some visits and we have a standard delete behavior. So block deletion.

Speaker 1

But in details it's very common that detailed data is not as important as main record and when we plan to delete main record delete data should be deleted. So our child detail data should be deleted without any questions like when we remove some contact, we are not caring about his mails and if it has a couple of males saved in contact communication so we don't care. We also want to delete them as well as addresses or some payment information or something else. So anything which is connected to the main record. If we decided to delete main record usually we are interested in silent delete of all the child parts.

Speaker 1

So here our reality visit is something that is not very important in case if we delete parent reality so we can set delete records from current object it means when parent reality is deleted, all connected visits will be removed automatically physically. This is performed by help of foreign key at the database and with option cascade on delete. So physical delete performs on the database level, not at application level. Unfortunately you can't handle you cannot handle such delete at application level because it will be triggered and made by database foreign key tools. But okay, so we have parent reality required special set for delete behavior. Good. This is our column for connection between visits and parent reality. Now we have to make some columns that will work for business part of us. So any visit will have date and time of visit.

Speaker 1

So we can call this colon visit date time. We can make some default value just to make it easier to set up new values and will be system variable current time and date Default value will be provided in case if you try to add the new record. And it makes sense especially if you want to change for example till tomorrow. But having similar time setting automatical value here will be quite good. So visit date and time it makes sense to make it required because having no visit time, it's definitely not good to visit data. One more thing we may know potential customer this is a lookup. We can call this potential customer. Why potential? Because we already have a real customer for our realty who asked us to help with sale or rentals.

Speaker 1

So potential customer is a person who probably will buy a house or will rent an apartment. So potential customer, we don't know who it is. So sometimes could be empty. And lookup is contact object which is a very typical way when we need to get some kind of personal reference. So usually we use contacts for this and for context selection. Maybe we need full scale selection window instead of list and using normal block delete option here. Also you probably will be interested in some very rare cases you may not need foreign keys at all. For example, you do something artificial like selecting data not from a real table but from some data view as a lookup. In this case you will not be able to construct a valid foreign key.

Speaker 1

So we have an option do not control integrity which means please do not create foreign keys by default. When you create a lookup column foreign key is created automatically. But if you enable option do not control integrity it means that foreign key will be not created. It's interesting fact that such option was used for created by you see do not control and for modified by. The reason why developers need it is performance. So created by and modified by is usually set automatically by system. You see system variable, it's a current user contact and usually system sets this value correctly.

Speaker 1

So such a column is usually not edited by end customer and it means that we have pretty no serious reasons to control it with the help of foreign keys only in case if you try to remove corresponding contact then such created by or modified by columns should prevent this data from Removal. But it's very rare case when you remove contact who previously was a user of your system. So generally we can say that in this example having foreign key is not so critical because value for this column is usually set by system and it's set correctly. So developers also realized that if we have no such option and let's imagine we have probably thousand of out of the box objects for our big base product like sales, service and marketing. So it will produce 2000s of foreign key.

Speaker 1

So for created by and modified by for each object, if you had no such option, we will have two foreign keys pointing to contact entity. So now let's imagine you have a database which pretends to be quite fast and comfortable for work for big company. And then you have a table in this database contacts table which has 2000 foreign keys on it. Each time you try to make any changes in these tables, such foreign keys will be invalidated and they will be checked

and it will be a huge performance impact on your database. And each time you do some operations like inserts and updates, such foreign keys are also checked and the contacts table will take a huge time for blocks that is necessary to perform low level database checks for work of foreign keys. So developers decided to sacrifice referential integrity here.

Speaker 1

And that's why they decided to make it with no foreign keys. For created by and for modified by and such behavior was inherited from base lookup. Oh sorry. From base entity. If you will search for a base entity in our packages, open it. You will find out that this column created by has option do not control integrity and the modified by also has this option do not control integrity. So this is made intentionally and because of performance. So maybe something like. Interesting fact, you may need this option in very rare cases when you create some artificial lookups based on something that is not a table. Like something that is a database view or even something more weird thing like a programmed stuff at the server side, not a database item at all.

Speaker 1

In this case do not control integrity will save you because otherwise system will fail to save your object where it can't find physical data based at your point on selecting your lookup data source. So here we have just standard situation potential customer based on contact. We have all standard settings and the only change was selection window. So full scale selection window selected just to make it more comfortable to select potential contacts potential customers as a context from big selection window. Okay, what else? Probably we need one more column here. This is comment. Let's make a text 200 and comments. Sorry. Yep, title will Be comment. So any visit may end out with something like we need to remember to write down and just to have some notes, maybe some decisions, maybe some ideas.

Speaker 1

So comment after a visit could be really helpful if you want to proceed with this realty. So we can have a comment as a text field, very simple, not required and just text value. Okay, so now it looks like our simple detail object is almost ready and we have to save and publish it. Clicking publish as you remember will automatically perform save first. Now you see it and then it will do publish operation and it also takes couple of seconds. Great, our object is ready. But what we need to do and what is a bit different from maybe you have some experiences older UI and maybe you have some previous creation experience. So Starting from version 8, objects are connected to page object connected with pages with a separate special setting inside of objects properties.

Speaker 1

So now you can see pages tab and we have to specify edit page for our reality visit. Obviously we don't have any page for this, so we can easily create with the add new page button we can create new page edit page for our detail we can specify select some parent template that will be used as a basis for our page. Our column set is not so big and impressive so we can just take some mini page example which will also look good if you have not so many columns. So I will select mini page and this mini page offers us possibilities. Okay, we can use some code, let's say, let's put some good looking code like realty visit page title Reality visit page. Okay, maybe we can call it form page just to make it traditional with realty form page.

Speaker 1

So reality visit form page Its parent packages realty template is mini page. Great, now we have to use existing columns. As you can see we can add new columns but we already have data model attributes already designed in our object so we can just show it up. So what we need here is visit date and time. Probably the first important column that we need to display. Second important column is potential customer and third one will be comment. Okay, so you may ask me and what about parent reality? It's mandatory. Why don't we show it here?

Speaker 1

The answer is that in case of detail management, when we add new data from the main edit page from main realty page, our parent realty will be already known and we should not allow to change it for user and we should probably should not even show it for a user. So we can hide this column so we cannot show it on the page. And in case of creating a record detail or in case of Updating new record or update an existing record or editing record. In all the scenarios we can set parent realty automatically. So we may not allow our user to change reference to a parent realty of an existing visit. And that's why I will not show this column for user at all.

Speaker 1

So it will be just hidden, it will be not shown on the page and our reality visit page will look like this. We can see why is there a red asterisk beside visit date and time and parent realty. It means the column is required system will not allow us to save data if required column is empty. Okay. And for parent reality we will guarantee that it will be filled incorrectly. So even if it's required, we will make we will take care about setting it properly in all possible scenarios. So we don't need to show it in edit page. We can show it and probably make it locked just to display and demonstrate that this visit is connected to some parent reality. But we will always look at this page from realty main page so maybe it's not necessary.

Speaker 1

This is very common for details and creature to hide parent column from the page at all. So we can save our page now. So save it fully. No business rules, no code, no any special settings. So try to keep it as simple as possible and when we did it so we can find this page in this list. So let's search for something connected to visits realty visit form page default page here means that it will be used for all scenarios to add, to browse, to edit data. So it will be used for all scenarios. And you can see it's possible to make a separate add record window in case if we need to do it in a very specific way. For example, when you add data you have to specify one set of columns.

Speaker 1

When you edit, you want to specify totally another set of columns. It could be used. And also we have an option for different pages depending on some specific field value. This can be helpful in case if your entity is quite big and complex. We have examples of such big complex entities in creatio and one of them is activity. And personally I don't really like this, but it turned out that activity is used for calls, for emails, for meetings, for to do tasks which you can create for yourself. So you can see a bit of different nature for the same entity. So you have one single section and many different purposes in it. Obviously for email we will have a different edit page than for a call and different set of columns, different display and different page design.

Speaker 1

And that's why we have this Option multiple pages based on specific field value. So for example, activity entity has a field value category which shows us is it an email or a call or a meeting. And depending on this column we can use different pages. And when you look at activities section, for example, you select an email, try to open its page based on this information system will show you email edit page. If you do the same for a call, it will open you call edit page for activity. But now I try to keep it simple. So we will have just one single page for all possible visits. Because we do it and we just need to get used with the details in general, we need to understand them well. So we have this option. This is mandatory.

Speaker 1

Okay, finally we finished and we can publish our object again. As you can see, we have to do it twice. First to create it, second to finalize assigning of a page for the object. Okay, great. Now our object is ready. We can close it. Now we

can see that in our package two items appeared. So we have realty visit page, we have object for realty visits. You can find some additional strange items called add ons. And sometimes we have add ons for pages and we can find add ons for objects. Add ons are separately saved pieces of metadata. You can try to open them, but you will see read only text in JSON which represents metadata. We have some a bit more user friendly way to read it. You see read only way and you can see property names.

Speaker 1

But in general this is only makes sense to base core creation developers who wants to check if everything is okay with metadata or not. In projects development and the end user customization, add on is just something that is saved in addition for the page or object. So we don't need to modify it. Attempt to modify. We should not remove it from the system. Let it be so it's just a separate stored stuff connected with page or an object. Good. So now we have our object and it's time to think how we can put it to our page and display corresponding data. So we can do this with the help of edit page button. As you can see, it's my favorite way to customize your system and other options.

Speaker 1

We have another options like finding your corresponding form page and open it from configuration section to be the same result. Or you can achieve the same result if you go to settings page physically system designer. Then you find application hub. Then you find your realty application. Switch to pages and find form page. So this same way for opening. Oh no no. Not visit reality form page the same way to open the Same page editor and I personally prefer to do it from section because it makes less chance to make mistake. And when you click on this to always open corresponding page and you should not focus on its name in the list of pages. Okay, so we opened our editor to design our page. Our task is to display details and details are usually displayed with the items called list or expanded list.

Speaker 1

I prefer to use expanded list and this is very traditional for creation because expanded list means you can collapse it and it will take less space if it's not necessary or you can expand it. That's why it's called expanded. I would say it's expandable when it's necessary to see some data from it. So let's use this expanded list component carefully drag and drop it on our page. I will use just free space in the bottom of this tab, but it's also possible to use additional tabs if you want to. You can organize your data stored in different tabs. It's up to you. So here we have expanded list and show part of it includes only settings like buttons and title and so on. And inner part of it represents data list.

Speaker 1

Of course we have to specify object first and as you probably guess, we have to find our real table visit object for our detail. It immediately shows us set of columns. So possibly we have to take care of set of columns. And I would like to show here first to be date and time potential customer, maybe comment and then create it on maybe just for reference. I would also display parent reality. Then I can hide unnecessary columns. And now you see how we should display our data. We can carefully shrink some columns. Okay, as you can see in Freedom UI system offers us horizontal scroll so we can add unlimited number of columns. We can flexibly design their widths and the horizontal scroll will show us. And Classic UI does not allow to do this.

Speaker 1

And there is no horizontal scrolls in classic UI and only 24 available columns to be displayed. So freedom UI is more flexible here. Okay, so we have this list of data, but what we need. How did you add the columns again? I just missed it. This plus add the columns and then you can select. Please note the order of how you click here matters. So if you click here and here you will see the same columns in corresponding order selected at your list. I first clicked on visit date, then potential customer, then comment in the same order. Columns appeared here and you can specify such columns in order to provide default column setup for your detail. It will be nice for your users to see your detail with

some preset columns so they will easily understand what it is about.

Speaker 1

And if it has some data, they will also see corresponding data and they will see what they expect. So here is our list. But also we need to do some additional settings in the right part. This is detail and detail must show us data only connected to his main entity record only for currently selected realty and not any other realty. So we have to make some filtering of page data. As you can see, this information tells us about that we may use it for example for detail filtering. So let's click on this plus and we have to construct simple filter which will be applied for all the data which you plan to see in this list. This simple filter is based one condition. Some column from reality object must be matched with realty visit.

Speaker 1

In reality visit we have suitable column parent real and system is smart enough to detect that you should use ID for main record for this. So here you see this filter by page data will be based on ID of realty and it must be equal to realty visit parent realty. That's how we will filter our data and show only visits for corresponding house, not for any other another house. Okay, this filter is mandatory, so you must do this. Otherwise you will constantly see big set of visits not related to your selected apartment. For example, no need to use static filters, but if you want you can apply there. And we have some options like row numbering, multi selection, allowed inline editing, inline insert. Okay, I can forbid inline insert and we will only use plus button to add new data.

Speaker 1

We can also program bulk actions for list. So when you operate with a list it's possible to select multiple records and then we have bulk actions. For example, you can program additional actions here and that's how you can do some customization and additional functions that will be running for selected data. They are called bulk actions. Bulk here means that you can select several records and then perform one single action for such records in order to so to process data more efficient. So we have element code here and it looks like our main list part is already set. All we have to do now is to find the small plus button. This plus button is designed to add new data to the detail.

Speaker 1

Of course its action will be open new record, it will be new reality visit object and system will find its page automatically. But here we have important setting which column values to set. As I told you, when we create new detailed data, we must take care about parent reality value. So we have to set here some default value for parent reality. Parent reality will have its default value. We have not so many options here. So value from another field and we will have parent realty id. That's the only thing which is suitable here. So when anyone clicks on the plus new realty visit page will appear. But parent realty for this data record will be already set up as ID of our reality. This is very important, otherwise we will not be able to normally create visits.

Speaker 1

So please remember this is filling of a column that is used for connection between visits and our realty. Okay, great. No any additional settings. So you just have to remember when you build your detail, you should set up your list and then take care about plus button. That's all. We can save our changes and test how our detail works. We need to reload page just to make sure we are not fighting with any kind of browser cache issue. Let's go to apartment in New Orleans. Now you see our detail. By the way, it's interesting this status of expanded or collapsed system remembers it and it remembers it separately for each user. So if I prefer not to work with this detail, I can collapse it and system will forever remember it. If necessary I can expand it.

Speaker 1

If other users prefer to expand it, they also may use this management and they can collapse or expand only necessary tails. So let's make a quick test of how our detail works. Let's try to add new record. You see default value of your visit date and time was set. Okay, let's set it for tomorrow the same time. Probably we do not know any potential customers, so let it be James Bond. Here is some trick. If you will click on this hyperlink, it will open corresponding contact page. So it's not what you plan. When we select data, it's better to click on the record but not on the hyperlink. So somewhere here double click. That's how we will select data in this column and comment some test comment. Save it. You see the data was created and parent reality was automatically set correctly.

Speaker 1

We can copy this data, save it, check it out. You see the parent reality is copyable. It's very important to keep it copyable. So in this case creating a new record was correctly copied previous parent realty. And obviously we can do it one more copy. We can edit any record by double click on it or by clicking open. And obviously we can remove data if not needed. So our detail looks very good and works correctly. Let's check that for other realty records. We don't have this data. So let's go to select and find another realty. Oh, we have something interesting here. Let's check Out. You see, our page is hanging now. Probably it's because of browser issue. I just updated it recently. So possibly we have some kind of browser related issue. And also we can close all unnecessary pages.

Speaker 1

Maybe it will help our system to work more correctly. Something unusual happens with my browser with my app. So let's check about its CPU and memory. Okay, I will reward the page fully. This is not a result of adding details, so it's probably some kind of crash but not related to our detailed data. So now you can see our apartments and visits for apartments were done. Okay, another house in Miami, another data. So we have no data. Let's make for a house in Miami some visit next month. We don't know who exactly will come. And we can save this planned visit. So depending on the record, you see corresponding detail values at our visits. You see. Okay, great. So now we successfully made our detail. Of course we have to take care about saving everything on disk.

Speaker 1

We have to submit our changes to version control just to finalize this step of our settings so we can like more confidently move on without afraid. Without being afraid of something can be lost in the middle. So I recommend you to do such comments in version control because it helps to keep all your history. And also if something happens with your environment, having such history will be a great help because you'll be able to recover anything that you developed. So in my case, now everything was saved. Let's commit it to version control. We have some data here. We have a lot of items modified. It looks like our section was never submitted to Version control. So it means like real tapp edit visits, detail added. Okay, so it would be a quite big commit for us.

Speaker 1

And we will sync changes which means pull and push to common repository. And now it's okay, it's done. So we can track our changes at our GitHub repository 7commits. You can see all the history, you see how many files were changed and so on. Each time we do this, you'll be able to track all the history and make control over what is going on here. Okay, so what's next? What we can do next with our details and with our pages you can program something in edit page of a detail with the same way how we did programming in our big form page. So this is. This page is produced. As you can see we have a button for edit page. We also will be able to do programming of its handlers or validation. So rules are the same.

Speaker 1

And also you can use business rules if you want to do some kind of filtering or simple fields hiding or showing you can use all the features. So you should remember that our page for detail is based on a separate client module that was created today. It was a reality visit page. So this reality visit form page, as you can see it's a separate client

module. We can even look at its code if interested. It has its own Vue config view model config and empty handlers, validators and converters. Maybe you have any questions? Maybe you have some like notes ideas or something. And in some implementations you may need to create a read only details. In this case you may hide new button, you may construct your object which will have no edit page at all. And you may also disable editing in list.

Speaker 1

So inline edit should be disabled. So read only details make sense when you have some data automatically calculated for it and such data is not expected to be added or modified by end user. Read only details are helpful, especially if you use some kind of integration with third party systems. And you can read data from third party data sources using for example some database to database connection if it's on site and then display such data as a list your pages. Okay, so that's how you may do such details and you may organize them in separate tabs of your page. Or you can put it down. There is no physical limit on how long you can scroll down. So you can put new items down this page and there are no limitations of number of rows. Let's try to move on.

Speaker 1

And possibly before we moving on and work with classic ui, I need to show you something important related to customization of existing pages and also important things about customization of about managing your applications. So let me explain what I plan to do right now. Let's imagine this is very typical by the way, this is a very common scenario. Let's imagine your business user asks you and says hey, you know, you already created your own sections. It's very good we do this. For example, we have realty section, we keep some data there, it's okay. But also inside of our customizations we need to do some changes at existing entities.

Speaker 1

For example, we have accounts which represent list of companies and we want to connect such accounts with some kind of third party website where we have a list of our customers and we need to make some connections with it. So we need one more extra column at our accounts section. And simply speaking, we need to add some columns into existing entities. We can do this easily with no code approach. Let me show you how it's possible to do. You can open account Page account form page. And then you can go and click on this edit page button. Let's try to do this. Let's see what happens. As we're running our system as administrators. So our supervisor user is a member of system administrators role. Now you can see account page design. And you may not.

Speaker 1

You may now see some very strange package automatically created. And system does not allow us to change it. And this strange package will be used. So it will be created by system automatically. And when we customize something, system will save our changes into this package. Okay, let's agree. Let's allow it to do this. So let's agree for this procedure. So our task is to add one more column to the account. So we will have a text value. Let's put it somewhere here. We can name our new column Website code. We can also provide its code for it. Website code copy sorry. Confirm text 200 great editable. Great element code should be unique. Okay, let it be developer friendly. So we made our new column. This is very typical. We did not use programming at all. This is only customization.

Speaker 1

But you need to know how technically this is done and where it will be finally saved. And how to for example, turn your system back to its previous step, previous stage. If you realize that such change was not designed, was not desired anymore. So when you change like this, you can save it. Also you can hide existing columns. Of course, in this case they will be not removed. You can move some columns. You can reorganize user page of existing section. It's also allowed to do. You can also add a lot of new components on the page, not connect to the database at all. So it's up to you how you do this customization. In my case, I try to keep it very simple. I just added one single column here. Okay, and then we save it.

Speaker 1

Now I should pay your so I attract your attention that we will finally receive new package. Our customization will be saved as a special type of item into this new package. So let's try to do this and then let's see what finally will happen. This is a fully valid flow that valid algorithm how to customize existing tools. You could also customize it the same by going to application hub, finding Customer360 app and then making changes for a page from there. Okay, so we made our new column. Let's close the page here, open it again. Probably we already will see it. Yeah, Website code. You can test some value here. As you can see, it was correctly. It was correctly saved. Now let's save it. Some warnings but close maybe let's hide some unnecessary columns in order to display important for us columns. Website code.

Speaker 1

Yes, it's saved correctly. So everything's good. We can even edit, inline edit. We can even do like this and save our changes. So then open it. You see, everything works just as planned. So we managed to create new column. It is displayed in our accounts section. Everything looks good. Now let's go to configuration section. Or maybe it's better to go here to Application hub because it will be more clear for you to see. Let's go to customer360app. Because our account edit page originated from this section. So it was designed in this section. Let's open it. Now we see some checks, some items. Our pages seems not to be changed. So we have the same set of pages as from out of the box. But when we look at number of our packages, we will see all three standard packages, but they are read only.

Speaker 1

And once we need to save some customizations system created automatically new package with this strange name and it created several items for it. And such items are not so easy. Let me show you properties. We have special type of object which is called replacing object. It's not usual object, this is replacing object. And also we have special type of item which is called replacing client module. You can see replace parent option is on and this is extremely important for you to understand and feel this difference. So now we are going very close to understand what is replacing object and why it is different from regular object and what is replacing view module or simply speaking replacing client module from any previously known and studied types of client modules. Possibly it's make it's better to start from objects here. So what is replacing object?

Speaker 1

Let's open it. Maybe we'll understand it better when we open it. First of all, it's interesting that replacing object it has replace parent option on which clearly states that this is not a usual regular object. Replacing object is used when your original object is present and your task is to customize it. But original object is not in your package and the original package is not editable. In our case, account entity was designed somewhere in base product creation packages. It was additionally customized in Customer360 packages, but they are read only if you want to change and save something. We have to use replacing object and save it into our package into new package. This package now is editable. It's very important. Also it's very important that in replacing object, parent object is mandatory. And now you see value here.

Speaker 1

Parent object is account and replacing object code is absolutely the same. This is important rule of creature architecture. So when you replace some entity, its name remains the same, you can change its title. Moreover, it was used in base platform bank software products where original account was replaced and sorry, it was renamed with a another name called legal entity. But physically it was still account entity and it's still the same table and the same logical name at program code. So only title was changed. So when you replace some entity, you can change its title, but you cannot change its code. Okay, and the code should be remaining the same. And personally I would like to make it not editable.

Speaker 1

As you can see here, this editor offers us possibility to change its code, but physically system will not allow us to save it and it will show us big red messages when we try to save such changes. So I think that this editor should not allow you to modify code in case if you work with replace an object. Okay? So code must be the same as original one. What we have as replacement, we have a lot of inherited columns. All the columns from original entity are shown here. We can slightly change them, we can change their title, we can modify their default value or make it required or not required. But we will not be able to rename original column, we will not be able to change its lookup property.

Speaker 1

And in general it's not recommended to change its type, even though system offers us such possibility to change the type in its subtype range. For example, if original type was text 200, then potentially you can extend it to text 500. But I do not recommend you to do this because you will face some additional unexpected difficulties. So it's possible, but I think not so feasible. So it's better not touch types of columns when you make replacing objects. So the main reason why you need replacing object, of course is creating your own columns like we did here. So this is our own new custom column website code. And actually this website code is saved into our report place an object. All inherited columns are obtained from parent entity. So we cannot remove inherited columns and we cannot edit any property.

Speaker 1

So we cannot rename inherited columns. We can only change its title. Our custom columns are shown here in columns list. And usually the reason why developers create replacing objects is need to add your own columns. Sometimes replacing object is created because you want to change some common properties like title of the object, like some behavior settings or maybe change of access rights settings. When you replace an object, you can modify access rights behavior change. And we have three types of access rights permissions related to objects in creation. Access rights set for general operations to Allow or disable selection, insert, update or delete operation for the object.

Speaker 1

In total we have record level access rights where some records can be editable and others will be protected and will be read only and some other records maybe will be only hidden and shown only for developers and also for some rare cases. You can also use column level protection of data where some columns are shown and other columns are hidden and other columns are read only. So three levels of access rights protection and when you make a replacement object you can also change such permission set. Okay, so the main reasons Generally we have two reasons of using replacing object, making your own columns or customizing and changing some properties of existing object. What is also important for you to know is that replacing object is saved into your own package and it represents some changes that you add into existing structure of your entity.

Speaker 1

This can be done. You can see it clearly if you go to configuration section. Let me explain. Let's go to all packages, search for objects and look for account object and make it strictly searched by equals condition. So only if object name equals to account it will be shown here. Now you can see we have some origin of it. Origin of account is somewhere in CRT core base. If we open its properties we will see that it was inherited from base entity base object. It's the same base object is title base entity, it's code and this is not a replacement, so no replacement. So this was original creating of new entity named account.

Speaker 1

But then after creating it in CRT core base package we had several reasons to do changes in completeness package, in email mining package and in some other packages including this weird package related to Customer360 app made recently. So you should understand that we have original structure of the account entity and then we have some add ons added to it. Finally, our last addition will be from our customer, so from our editable package and it will be like a final layer of all the settings. As you may guess, the order of application of such layers is important because if this one is applied later than ours, so this one will declare final values of certain properties that we need. So the order of applying of changes is really important. And usually this order follows package dependencies Diagram.

Speaker 1

So if our package is somewhere at the bottom of the structure, our changes will be added the last and it will be no problem for us to specify final properties of our object, for example disable some access restrictions or change some column titles. Final structure of account is formed by its origin plus all the replacements. It's interesting to mention that at the database level we have just only one data table. Let me show you. This is our set of databases D1 database which will work for now we have account data table. We can extend and see its set of columns. Now you see for account table we have a lot of base product columns and the last one is our website code that we added recently. So replacing object does not create any new table.

Speaker 1

Replacing object uses existing table and adds new columns there. If you will remove replacing object from your system logically your new column will disappear physically. It will be present in the database as a kind of leftover. But it will be not so critical, not so important. And I think that end users will notice it at all. If you will recover your package again, you will still resurrect this column logically and it will be available at application. So that's how replacing object works. And you can use it to customize any existing part of creation objects, including some base objects like base entity and base lookup. So you can customize quite flexibly, even some, let's say top level objects that are used for many of inherited child items. So you can use replacement for any existing object.

Speaker 1

And Adrian is asking very interesting question which I plan to move on here. So you work with some account page and you would like to keep your settings saved in a proper place. Yes, Adrian, I will show you now and I'm just going probably not so fast with explanations here. So now we have Customer360 app and for this app system created editable package. And we have some contents in this editable package. We have some customizations made especially for this app. Let's imagine that you have more apps like you have sales automation app when you add. So for example, you have case management app. It's a base product for service automation. You have some lead generation, some opportunity management. So let's imagine you have several apps and you want to customize most of them.

Speaker 1

By default system will create separate editable package for each scenario, for each case when you want to customize some existing app. But obviously you may not want to work with such many packages. Also you may not want to work with packages with ugly names. Let me explain what I'm talking about. About let's reload our data, go to all items. And now you see package with some ugly name, usr CRT customer app and then some strange unique suffix added to it. So the same name will appear on our discord. Probably already appeared. Let me check it out. So I expected to see here. Yes, the same strange name appeared on our disk very soon this name will travel to version control and it will be not so comfortable for other developers to see such a strange package.

Speaker 1

So your question and Adrian question is asking about is it possible to use your own package instead of such ugly stuff that was created by system? And the answer is yes, we can manage this. This is not obvious and you have to perform additional steps to enable such management. So let me show you how you should do this. Let's go to lookups or we can go to studio lookups. Okay. And then let's search for recently added lookups first. Sorting by date of creation. Okay. Then we can register special object as a lookup. This object is named something like packages. Yes, this name is package installed application and this is quite not obvious. I only knew this from my colleagues. So this is kind of a technological secret and I think it's not very easy to find. But if you know that it exists.

Speaker 1

So you will enable it of course, package installed application and we will provide the same name for our lookup and try to save it. Okay. This is only developer tool so we do not need to transport such registering of package to test and production. We can open contents. And now you will see all installed applications, all existing packages and you will see settings for all packages. Are they primary for a certain app and are they used as a default current package? Current package means editable package that is used to save new customizations there. So let me show you how we can do some tricks. Now we have customer 360 app and we have this trench package is used as a primary and current package for this app. So any new customization that we do in customer360 will fall into this package.

Speaker 1

We don't really want this, so we want probably to remove this package at all. And maybe we would like to use our existing new package which is a dev package. We have it Dev Classic package. So we would like to use dev Classic package for this. So we can try to do this replacement instead of this. We can clear and then search or maybe just type dev Classic package. So we can put our existing package as a target default package for Customer 360 customizations and confirm our decision. Of course we will have to take care about package dependencies. Now our dev Classic package has no dependencies, but it will appear so it will receive it soon. Let's go to our application hub. Now try to customize customer360. Go to advanced settings.

Speaker 1

And now you see our dev Classic package is connected to this app and we can see properties of this package. Let's check. What about dependencies? You see that System automatically added Customer360 app as a parent dependency to this package when we made changes, when we customized this stuff. So we did something like this. Oh, something like this. Yeah. So we did this setting and it was important. So now we have automatic changes in package dependencies. Okay, great. So you can see Dev Classic is connected to this app. And probably the last step that we will need to do is we have some customization made in this ugly package and we can transport it to our Dev Classic because we don't want to use this package at all. We want to get rid of it.

Speaker 1

And we will continue customization of Customer360 in our classic package. Andreen, is it what you were asking for? Okay, great. Thank you, Adrian. So now we have multi action move to another package. You should be mentally ready that this movement takes significant time because system performs a lot of checks and sometimes it fails because of missing dependencies or some other technical reasons. But in general it's quite safe too. So you will probably run it fully correctly if everything is okay with dependencies of target package. So now I'm trying to move items created in my ugly package into Dev Classic. Then I will I plan to remove my ugly package at all from my system. It will be empty and I can safely remove it.

Speaker 1

As you can see, this transfer takes significant time because it looks like system performance performs a lot of

checks of items that we process and it makes sure that we have no other connections to this item. So it does something like probably some excessive work. But okay, it makes it successful. Now you see, our customization traveled to this Dev Classic package. Now if we go here, you see our customization should be present here. Probably it's because of caching. So let's reward. Yes, now it's here. This information probably will. Let's reward this part. Oh, it still remembers of this connection. So we can remove now this package and this line also will be disappeared. Let's let me show you. So this ugly package, now it's fully empty. We can remove it, delete it fully. Also removed from the apps package. Remove takes time. Okay, please.

Speaker 1

Now it's empty. It should not take a lot of time. Yes, successfully deleted. So we have no ugly package anymore. We have some correctly created packages, good names. We use this good named package as a target package for our customization of Customer360. Now you see everything looks good. Our changes that we made are saved in a proper place. So that's how you can manage connections between packages and applications. I agree this is not transparent, so let's close open this register. Okay. Package installed application. You see no ugly package at all. We see our dev classic as a target for this app for realty. We have only one package, so you can easily add the new data here if necessary to connect some packages with some applications, you can do whatever you want.

Speaker 1

Of course, you should be quite rational here without obvious mistakes that you can prevent because this is developer oriented tool. If you intentionally do something incorrectly, it will not save you and it will try to do what you're asking for. So you should consider yourself before making changes here because it's like architecture but you can manage. Thank you Adrian for your questions. Okay, it's time for us to have just a short break, probably five or seven minutes. And Adrian tells us about the question. Customize account Page and Freedom UI generates new package. Yes, it was in. In January 2024 and the answer was from Ryan Farley where we had probably no such tool. So he advised us to do something like movement of items and then just use this loop up. Yeah, yeah. Thank you. Thank you, Adrian.

Speaker 1

Okay, thank you for all of your active questions. Let's make a five or seven minutes break, we'll refresh a bit and then move on with next topics. Thank you.

Speaker 2

Hi.

Speaker 1

Yes, hello.

Speaker 2

I have a. I have a small question I have. Yeah, so I said we use the two lookups.

Speaker 1

Okay.

Speaker 2

First one is for country and second one is for city. Right? So in the drop down button we have one plus icon is there if you remember.

Speaker 1

So what about which lookup you're interested? Is it the main realty or.

Speaker 2

Yeah, main reality. We have country and city. Right? Country, drop down and city.

Speaker 1

Drop down type.

Speaker 2

No country and city.

Speaker 1

This one. So I can't hear. I can't hear properly. Country and city. Okay.

Speaker 2

Yeah, yeah. So we have a plus icon over there, right? Adding. So what is the use of that? Or can we hide that button if possible?

Speaker 1

So you ask about if it's possible to manually create new city for this.

Speaker 2

Okay, Can I, can we hide that button? Disable that button if possible.

Speaker 1

This. Yes, sure you can. I just forgot to do this. Yes, let me show you. We can go to edit page. We can find corresponding lookup country and there is an option to allow add new value so we can disable this. And for city we will also disable enabling new values. So just tick off save it and we will only be able to select from the list without possibility to add any new countries. Close open. Now you see some selection. Now you see only available columns. Thank you. Rakhi states. Oh, we have New Orleans, no corresponding city. Okay, would it be. Thank you for a question. Any other questions, feel free to ask because now I plan to show you something which will be a final step about user interface.

Speaker 1

And you should know that we are working with Freedom ui, but maybe you will also need to work with some classic UI old projects. So you need to get some basic understanding of how classic UI can be managed. And before proceed with classic UI customizations, we need to make some preparation that is necessary and very important for classic ui. So in order to create new section in classic UI you need to prepare target package. We already did it. Here is our dev classic package. But also you need to tell creatio this package should be used and we will go to system settings, then we will find a special system setting named current package. Open it and by default you see custom here as a current package. Rakhiv is asking can we change the color of collects icon?

Speaker 1

Oh, I'm not sure because it looks like this is hard coded thing. So possibly you may find somewhere in CSS value for this color, but it looks like this is an image designed somewhere in the app. So it was not supposed to be customizable, I think. Let me show you with CSS it's not possible. Probably it's because of this is an image. Okay, let's try to understand. Let's try to do inspect. Then you see we have so inspect some marker, some kind of div and what else? It looks like we have some image. As you can see such images are hard coded, but you may find where they are saved. So possibly you can replace such images with something else.

Speaker 1

You see the current source, so you may try to replace actual storage of this image and you may design your own and that's how you will finally make corresponding change. Rakhi, is it clear? Of course it's not recommended way of customizing creation by changing of its core files, but physically you can try to replace this file with something else and check it out how it works at your environment and then you can if it's on site, you can also get this file to the other, let's say test and production environment, so it may work for you. You probably will not be able to apply such change at cloud conditions, because cloud engineers will definitely will not agree for changing files somewhere in core part of Creature and it will be not saved into your package.

Speaker 1

So this is part for core which is not designed to Be changeable by configuration changes. But if you really want, you can replace files on disk with another ones and that's how you can do this. Thank you Rakhi, Very interesting question and let's move on. So I just wanted to explain you that in case if your plan is to use classic UI customization, then you will have to take care about current package system setting. By default it is set to save

settings into custom package and we plan to save it into our dev Classic. We should do this before starting of any customization. And also take care this system setting is not cached. Cached settings are loaded at the start of user session and cached settings if you change them. So you will need to fully apply changes.

Speaker 1

And if you change cached setting, you will have to log out and log in order to read changed values properly for your user session. In the current example, current package is not cached, so we may change it without logging out. Okay, great. So we can change save it then. One more tip that you need to know before starting Section Wizard. We may experience difficulties because of recent changes in object model without compilation without application restart. So we will need to restart our app in order to guarantee normal work of Section wizard. Especially on save object operation of Section Wizard. As you remember, restart of the app now can be done quite easily with maintenance tools made by our Swiss partner. So restart application. This is necessary to normalize our server side cached data about object structure.

Speaker 1

If you recently made changes in your object and even published it still may keep some incorrect settings at application server level cache. So we need to do this restart. In this case everything will work. Okay, good. Now we go to settings. Then we find item called Section Wizard. And this tool is designed to create new classic UI sections. I will quickly make some simple example not as complex as we made for Freedom UI just to demonstrate you such tool. So I will do it fast. May you may not reproduce all the steps after me. So it's up to you. Would you like to do this or not? So I will make some new classic UI section. When we do this we have to provide its title. So let's call it Realty Classic.

Speaker 1

And we have to be very careful with the code because code will not be able to change if we finish its input. And that's why we have to type it correctly from the first attempt. Realty Classic. So this is our code. Once we move our focus out from this control, you will see system will think for a bit and then we will not be able to change this code. So if you made mistake. Then you only cancel and run section wizard again. Workplace. Okay, let's put it to the same studio workplace and background icon. Let it be green so it will be our icon for the color of our section icon. Okay, Great. Then we go to setup page. We need to do some minor settings. I'm still a bit afraid of what may go wrong when I save my section wizard section.

Speaker 1

So I will only change something like a name column. I will provide some minor change copyable setting, no any other settings. So I will just go back and try to save my section in case if it saves correctly, I'm happy and I can continue customizing of my section. But in case if it fails to save and it will be obvious that I need some more steps to prepare my system, then I will not lose time for setting up columns. So saving objects is the most critical step of this procedure. If it goes correctly, then we will successfully continue our customization. If it fails now, it's correct. Yes. Great. So if it fails, we will make something like discovery of what went wrong. But now in our case it looks like everything is okay. Great. So we saved our section correctly.

Speaker 1

Now we can edit it. And editing a section is much safer than creating it. So we can go to edit page. Now we can customize it. And I will quickly add similar columns as we had for Freedom ui. So decimal column drag and drop to be our price. You see a bit different user interface, but the same sense. We add new columns, we provide title code, no element code and just save. I provide area I can use lookups. As you remember, we made a lookup for type of our reality. We can call this column USR type. The reason why we use prefix is the same. So system asks us to keep prefixes for our customized stuff bus telling that he has two leave no problem. Thanks you. Thank you. Thank you. And you will watch videos. Great. We will finish soon. Don't worry.

Speaker 1

Today's Friday and probably tired a bit. So copyable value type based on existing lookup realty type or Classic UI Drop down list block behavior, no redundantly, no hiding titles. Everything is okay here. Save it. Great. I will show you how we can make new lookups in Classic ui. No existing lookup so we have to make offer type. Code is USR offer type and no existing lookup. So we will create one and as usual we'll provide title for this new lookup Realty Offer type Classic ui. So this is how we provide title and code for our new created object that will be inherited from baselookup and register it as lookup by section wizard, drop down list, standard behavior for delete. No any other options. Okay, great. And finally we can make some kind of text just to make comment.

Speaker 1

By the way, I will show you some trick which is supported in Classic UI but is not yet performed in Freedom ui. Comment here in their name is comment. But what is interesting here we can make a special option called multiline text. You will see what will turn it into the column. So multiline text has a very interesting feature which is not implemented in Freedom Live. I will show it for you. Multi line text. Obviously it's designed to make possible multi line values here to be said, maybe it also makes sense to have 500 just to make sure we have big enough value. But multi line text has also very interesting property. You will see it soon if necessary we can also add an object for detail and register detail. I will try to keep it simple.

Speaker 1

We have business rules here similar to Freedom UI and also we can program in JavaScript code if necessary. And let's go back keep it simple and save the changes. So I want you to see how Classic UI section looks like. In Classic UI we have much more objects created for a section. Now you will see in configuration section close properties of package, probably reward it. Okay, saved correctly. Now I reload section with configuration so we can go to dev Classic sort by date of modification and you will see how many new objects were recently created. Let me show you. We have objects for file for folders for in folders for tags feature and also we have object for main Classic reality section and we have an object for our lookup created.

Speaker 1

So we have much more objects for Classic UI section because such things as tags, folders and files are saved in a separate dedicated objects. And we have more tables for this for the database. And in Freedom UI such tools are incorporated into commonly used objects for tags for files for folders and it makes it more efficient storage. Okay, so let me show you how Classic UI section looks like. We will have this realty classic section but it probably will fail when we try to open so we have to reload because of Classic UI is much more sensitive to cache issues. So you should always reward your page. If you do any client side changes in Classic UI you see it's still faces some troubles. Let me show what's happening there. Or maybe we just have to wait. No, it doesn't look good.

Speaker 1

Okay, let's go to another page. Go to reality Classic page. Yes, we have some minor issues here, but in general it works. Okay, Try to add new record. Here you can see how Classic UI edit page looks like. I need to tell you that in Classic UI we have a totally different background frameworks used for user interface. In Classic UI we use EXT framework EXT versions. So you see it's Quite old, probably 10 years old framework and it's quite old but it works. And it's a JavaScript framework which keeps controls and some UI stuff. And in Freedom UI we use Angular so classic UI uses EXT and TerraSoft core framework. Freedom UI uses Angular framework so it's totally different and module structure is similar but still we have some serious differences. When we create new record we didn't assign any default values so far.

Speaker 1

So okay, let it be Paris Ren. The type is apartment. Offer type is not filled in so far. And here is what I wanted to show you. This is voice input and speech recognition piece. So you see it works only in Classic ui. In Freedom UI we don't have it so far. We don't have it in Freedom UI so far. And recognition quality is quite high as you can see. So you may use it for Classic UI implementations. And I think that in Freedom UI system developers should develop something like this and enable it. In Freedom UI now we don't have it and offer type is empty so we have to probably take care about offer type for Classic ui.

Speaker 1

This one it's empty so we can simply add sale and rental for our Classic UI and also we can bind all data to save it into our dev Classic package. So this is important step because such data also should travel to our test and production. Now if we go to Reality Classic you can see that offer type can be selected and we can save it. Okay, looks good. Some minor steps, minor preparations. For our section we can go to Workplace Setup Studio. Now we have our section for Classic UI quite far in low rows we can put it upper. Maybe we can also bind the contents of this workspace and at the same IoT save it successfully. View details Three items were created but you remember. First one will be information about workplace. Only one data row, nothing changed.

Speaker 1

This one will be about position of a module in Workplace. Let's check Reality Realty Classic. Great. Third one will be about access rights. That's okay. Rakhi asks about voice languages. It depends on browser. And also I need to tell you this voice recognition feature is browser dependent. For example, if I try to do the same in my Firefox. Let me show you my Firefox here. It still doesn't work in Firefox Let me quickly show it for you. It works in Edge and in Google Chrome recently. So I asked some colleagues. It looks like it works in Safari as well, but unfortunately it doesn't work in Firefox. Let me show you Firefox here you see no support of this feature.

Speaker 1

So it's also browser dependent and you should check and test it before you advertise this for your customers because it's a browser dependent feature and it looks like it depends on HTML5 elements or something like this. So some browsers does not. Some browsers do not support this and that's why you see it's not in any browser. Okay, thank you, Rahi. And what about languages? So so languages now let me remove this part now. So languages are supported and I successfully used other languages to normally detect it. You should use change. So if you change your user profile and change to another language, then your voice input recognition will also support chosen language. So yes, it supports other languages. At least two more languages which I tested. It worked well for me. Thank you Rakhi for your question.

Speaker 1

So finally we had an example of Classic UI section. We can set columns for our section list set fields to display in Classic UI we have different approach for setting up columns. So we can provide column settings type, maybe you need price offer type. Okay. And finally we can have a menu comment. Okay. And then we can save it for all users. When you save columns in Classic UI for all users, then it's possible to use data binding tool to help us to remember such column setup in our package bind column setup for Classic UI save it into our target package. And by doing this we can transport setting of columns for this list to other environments. So it will not travel automatically. Only if you make corresponding data item. It will save it and you should do it default settings.

Speaker 1

So you should save for all users and such default profile will be saved as data item to our package. You can find it here reload and that's how we can change and transport Classic UI columns setup to test and production. This one

is saved in sysprofile data at the same with details. Okay. Probably today's Friday and it's enough for today. And I would just want you to see a couple of images of what client modules in Classic UI looks like. So let's focus on clients modules only. Now you see we have section page and we have edit page. This one section page looks like a bit weird. We have a not so good looking name of it. We can try to modify it. Let's try to fix it so we can rename Quest 6 something section.

Speaker 1

But we should do the same code here and here. We should do it very carefully. Save it. Now we will have better names for our client module. Also it will be renamed on disk in our dev classic package. You will see realty classic section. Yes, this one renamed and this one was also renamed. So you see we have full correct renaming on disk as well. And after doing this I recommend you to instantly check it out if your section works well or not. Because now you see we have previous URL. Probably it will fail now and we have to probably simply go back somewhere like this. Go to classic UI section. Now you see another URL. So now it works. But we still miss columns because of changes and previous column setup was set for previous name of the client module.

Speaker 1

That's why we have to do it once again. Okay, Price type, offer type, maybe comment. Then we save it for all users providing so called default profile. This is important. After we do this we can bind columns to our package creating one more data item in our classic section data was bound successfully close it. Now we have to review what we got in our package in order to remove previously created unnecessary data item here old one is not needed anymore because it represents old values, old settings which will be not so practical. You see, because we renamed our module and old name was remembered here. Even if we actualize it, probably we will see there is no data for this. Oh, it's still here. But we do not need to keep such data in our package so we can remove it.

Speaker 1

It doesn't represent any useful data for us. This one is correct data with new client module. So when you rename client modules because of ugly names, take care about quest API column settings because you will probably have to review them afterwards. Okay, so final quick brief look at our client modules in Classic ui. So in classic UI section page looks like this. When you open it you will see this code. You will see define word, module name, dependency array, factory function and some body. So this is a body of our function which mainly returns an object with some set of settings. In our case it's almost nothing. So we have no any customization here. Classic UI pages are also inherited from some parent templates.

Speaker 1

That's why we have a lot of common functionality and no need to implement this functionality each time in each separate section. So our functionality is now in parent one and all the buttons for classic UI section, like new actions, view buttons, grid folders, feature tag, everything is inherited. That's why we do not need to customize it and that's why this code is so small. But for page for edit page code you see much more significant difference. So you can see set of sections module name, factory function returns a bit bigger object and you can see set of columns here. So they are different from Freedom ui you should focus on Diff first because diff shows you information about user interface changes and each item in this diff array represents one control as the same as we had in Freedom ui.

Speaker 1

So if you need to change something you can do it carefully here or better to rely on Section wizard because it will do it better for you than you do manually. And in Classic UI Section wizard doesn't allow you to use all available components, so you should be so in general Classic UI programming is harder because you can do some additional components and display them, but Section wizard does not support it. So in some cases, for example you want to add a button, you will find at Academy articles this definition of a button, put it to your code, provide corresponding handler and methods will be used as a place for your JavaScript functions to handle events of your controls. So in

Classic UI you can also do almost the same programming as in Freedom ui. If really needed, just tell me.

Speaker 1

I can show you a bit more examples, but I believe you will not focus on Classic UI so hard and this is just for brief review for you. Okay, that's all for today. We just briefly mentioned Classic UI and we probably need to pay some attention on migration from Classic UI to Freedom UI because it will be very typical tasks for most of our customers who started questiqi Solutions and unfortunately such migration is not automated yet. So today's session is over. Thank you very much for your time, for your patience and questions. Next week we will start with some migration questions from Classic UI to Freedom UI and we will move on with server side programming. So we will do some server side examples, we're writing C code, we will do integration with Visual Studio, working with File System and so on.

Speaker 1

As a final step for today, it looks like I unnecessary caused it. I have to remember everything we did in File System of course then I will submit it to Version Control just to keep all the history. And any of you who wants to keep how our session proceed, you can also watch our repository at GitHub. Thank you for your time. Our session is over. In case if you have any questions feel free to ask. Our session is over and goodbye. Thank you, have a good weekend.