i-net Clear Reports 2016

Ad Hoc Reporting

Usage and Customization

i-net software

1 Content

Ad hoc reporting is a powerful feature of i-net Clear Reports which allows even unexperienced users to quickly create meaningful reports. As the operator of an i-net Clear Reports server, you're able to specify the ad hoc layouts and datasources or dataviews available to the users as well as the appearance and features of the ad hoc user interfaces.

This guide describes how to configure an i-net Clear Reports server to run ad hoc reporting, how to create the ad hoc layouts and data source files or dataviews for the users. Additionally it gives you an introduction on how to modify the user interface of the ad hoc reporting web front-end.

2 Terms and Definitions

There are several terms which have a special meaning in the context of ad hoc reporting.

2.1 Ad Hoc Layout

An ad hoc layout is the basis for any report in ad hoc reporting. The layout defines the style of a report and the elements, which can be set and modified by the user.

2.2 Ad Hoc Report

The ad hoc report is what the user creates by selecting an ad hoc layout and setting it's elements. The ad hoc report does not refer to the result which is generated when then users renders the ad hoc report!

2.3 Dataview

A Dataview defines a view on a data source. Dataviews hide away the complexity of database access, filtering and formula definitions and will present only a plain list of fields to the user. Data Views are very useful to increase the usability and security of your ad hoc service.

2.4 Page

The ad hoc reporting swing- or remote interface presents the properties of an ad hoc layout in form of several pages. Each page has a special purpose like selecting an ad hoc layout, defining the fields to display or setting up a chart.

3 Configuration

This chapter describes how to set up ad hoc reporting on an i-net Clear Reports report server. Please read at least chapter Security since ad hoc reporting may at worst expose the datasources of your server. All adjustments described in this chapter are made in the configuration manager.

3.1 Layout and Dataview location

Ad hoc reporting has the ability to obtain the ad hoc layouts and dataviews from three different locations. It will scan the specified location and all of it's subdirectories for ad hoc layouts and dataviews. Please note that the subdirectory structure is as well the grouping of the layouts.

3.1.1 Internal

Internal sets the location to the layouts which are installed by the setup. These layouts are located in the subdirectory 'templates' in your installation directory. This option is active by default since the internal layouts are always available.

3.1.2 External

Switching to an external location requires to set an explicit path. This location will than be scanned for layouts and Dataviews. If custom Ad Hoc Layouts should be used it is recommended to use an external location instead of copying the layouts to the internal 'templates' directory.

3.1.3 Repository

With this option set, ad hoc reporting will scan the current repository for ad hoc layouts. Since there can be only one active repository at a time, ad hoc reporting does not scan all configured repositories. An optional subdirectory can be set to restrict the part of the repository which is scanned by the ad hoc reporting.

Please note that this option requires a Plus License.

3.2 Storing

The storing option enables the user to save his ad hoc reports on the server. To enable this feature, an authentication mechanism is required on the server. For more information about login please refer to the security guide.

3.3 User Upload

The user upload feature allows your users to upload their own data sources in form of a CSV file. These files may be for instance database dumps or some spreadsheet program. Please make sure to set the limits for the upload feature.

3.4 Security

Ad hoc reporting enables users to access data sources on your server and execute custom reports. By default, ad hoc reporting is enabled but protected by the remote interface password and the IP-Filter. If you want to allow your users to use this feature a complete set of security settings should be applied.

3.4.1 Restricting the Access to the Ad Hoc Reporting Interface

The ad hoc reporting feature can be restricted for certain users by activating and setting the ad hoc reporting permission. This can be done in the 'Permissions' category in the configuration manager.

There are two permissions which are relevant to ad hoc reporting:

- **Ad Hoc Reporting:** Enables the ad hoc reporting and the XML Interface and thus allows the user to use ad hoc reporting in the remote GUI and as a Java Applet.
- **Ad Hoc Data Sources:** This permission enables the user to use the data sources registered at your server for ad hoc reporting. This permission is critical since the user may use ad hoc reporting to read data from your internal data sources.

If the permissions are activated, these permissions have to be set for every user who is allowed to use ad hoc reporting. Every not administered user or permission counts as forbidden. Please set the permissions for the guest account as well, if the guest account is activated.

3.4.2 Restricting the Ad Hoc Layouts and Data Access

If the ad hoc layouts and ad hoc dataviews in the repository are visible to the user, can be restricted as well. To do so, simply configure the permissions in the repository browser. Now you can define the permissions for every ad hoc layout/ ad hoc dataview and user/group.

Please note, that ad hoc reporting requires only the execute permission to unlock an ad hoc layout for a user.

3.4.3 Using Data Views Instead of Datasources

As a guideline, the access to the servers data sources should only be granted to 'normal users' if your i-net Clear Reports Server runs in a protected Intranet environment. In any other case ad hoc dataviews should be preferred.

An ad hoc Dataview acts as a broker between a datasource and the field, the user is allowed to use. It completely hides away the datasource and the internal structure of the database. Furthermore it allows the administrator to define additional filters and use the formula language of i-net Clear Reports to modify data on the fly or create artificial fields for the user.

For a detailed description of ad hoc dataviews have a look at chapter 5 Ad Hoc Dataviews.

3.4.4 Limit pages and CrossTab

Ad hoc reporting enables the users of your server to create and execute reports which may impact the servers performance.

There are three options on the "Performance" tab which are relevant especially for ad hoc reporting:

- **Stop After Page:** Limits the number of pages, which will be rendered. This will prevent long running reports.
- **Database Records Limit:** Limits the number of records read from the database. This prevents the server from running out of memory.
- **Crosstab Cells Limit:** Limits the overall number of cells in a crosstab. This prevents the server from running out of memory.

Please note that these options are global for the server. Setting the page limit for instance will restrict every report executed on this server whether or not it's generated by ad hoc reporting.

4 Ad Hoc Layouts

The ad hoc layout is the basis for any report created with ad hoc reporting. It defines the elements, which can be modified by the user, their formatting and the design elements of a report. Layouts may as well use static elements and fields which are fixed in quantity and / or appearance.

The types of elements which are present in an ad hoc layout defines which pages will be available on the GUI to modify these elements. For instance, if there is at least one group in a layout, the group page will be available for this layout.

This chapter describes how set up an ad hoc layout using the i-net Designer.

4.1 Types of Ad Hoc Layouts

There are two types of ad hoc layouts available:

- **Real layouts:** Defines the position and appearance of report elements but don't have a data source yet. These layouts require the user to set at least a data source and configure one of the primary report elements (columns, chart or cross-tab) to receive a suitable result. This type of layout should be preferred when running an ad hoc reporting.
- **Ready reports:** These reports are basically normal reports created for i-net Clear Reports. These types of layout can be executed right away without any further modification. The user will be able to apply minor adjustments to the elements of this report like adjusting the filter.

4.2 How to create an Ad Hoc Layout

The condition for an i-net Clear Reports report to be an real ad hoc layout is the usage of the LayoutDatasource. This data source will be created by the installation process. It does not refer to any database. Instead it provides template fields which will be replaced by real database fields at the time the ad

hoc report is executed.

So to create a basic layout, simply start the i-net Designer, create a blank report and set the LayoutDatasource as it's only data source.

Please note: if there is any other data source than the LayoutDatasource, the report will be regarded as ready report.

4.3 Columns

To create a table style layout, at least one field has to be added to the details section of the layout. These fields will later on be replaced by the fields chosen by the user. The value type of the LayoutDatasource field added to the details section is irrelevant.

There are three types of column layouts available, depending on the page layout and the number of fields added to the details section.

4.3.1 Variable columns

To allow a variable number of fields per table row in an ad hoc report, add exactly one field to the details section of the layout. This field will be copied at runtime for any field the user adds to his ad hoc report.

The distance between this field and the left border of the details section will be the left gap of all columns of the resulting ad hoc report.

If you add a vertical line to the right of the column, the distance between the right edge of the field and this line will be the right gap of all columns of the resulting ad hoc report.

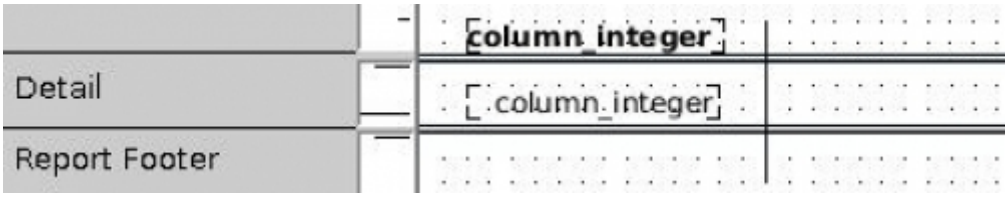

Figure 1: Variable Columns Layout

Any other design element like a horizontal line or a box will not be copied or modified. It will appear in the resulting report at the designed position.

If a user now adds for instance three fields to his ad hoc report, the rendering result will have three columns, each one with the designed gaps and separator lines between them.

Figure 2: Variable Columns Layout, render example

As you can see, the width of the field itself is scaled to fill up the remaining space between the gaps. So the width of the designed field is not relevant for the result, but there is a minimum width for all types of fields.

To use the automatic column name, place a text field with exactly the name of the data field in the section above the detail area. This is what the i-net Designer will do automatically by the way. This field will be replaced by the name of the column at runtime. In our example the bold text field with the content 'column_integer' is replaced with the column names 'ProductName', 'Price' and 'Size'.

4.3.2 Fixed Columns

If you add more than one field to the details section of a report, the ad hoc report uses fixed columns.

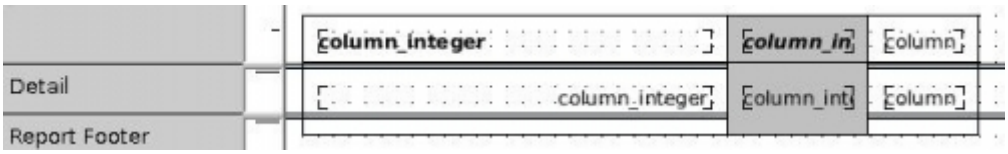

Figure 3: Fixed Columns Layout

Fixed column layouts do not scale or copy fields horizontally. Instead the layout will be used as is.

ProductName	Price	Size
Douglas' Red Apples	1.86	5.1

Figure 4: Fixed Columns Layout, render example

If the user selects less fields than defined by the layout, the remaining columns will be left empty. If the user selects more fields than defined by the layout, the details section will be copied vertically to place all columns.

ProductName	Price	Size
Douglas' Red Apples	1.86	5.1
Apples	9.00	

Figure 5: Fixed Columns Layout, render example with too many columns

Please note that the header is not copied. So if the user selects more columns than defined by the layout, the table header and the cell contents wont be consistent anymore.

4.3.3 Mailing Label

The mailing label mode is used if the page layout is set to multi-column or label. In that case the available space for the layout is limited in width and height.

Mailing label allows fixed and variable columns as well, but the fields will be placed by another algorithm at runtime.

Please note that there is no automatic column name for mailing labels.

- **Variable Columns:** For mailing labels the columns will be copied vertically instead of horizontally.

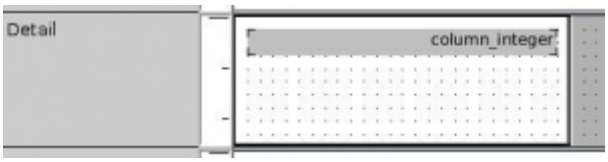

Figure 6: Mailing Label Layout

If there are to many columns to fit into the height of the label, the label will be divided horizontally as well to create several columns and rows.

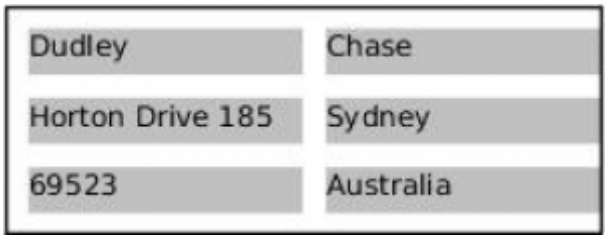

Figure 7: Mailing Label Layout, render example

- **Fixed Columns:** The fixed columns layout algorithm of mailing labels works like the normal one. It only duplicates the detail area vertically which will usually fill up the complete label by a few fields.

Any field that overflows the height of the mailing label will be invisible.

4.4 Groups

Grouping will be available to the user, if an ad hoc report contains at least one group. In case of a real layout the user can add or remove groups, in case of a ready report the user will only be able to change the defined groups.

4.4.1 What happens when the user adds a group?

If groups are added to an ad hoc report, first of all the existing groups of the ad hoc layout will be filled. So if you've designed three groups and the users adds three groups the result will display the three designed groups and their layout.

If the users adds more groups than available in the ad hoc layout, the last group header and footer will be copied including it's summary fields and name elements.

4.4.2 What happens if a group is removed or left out?

If a group defined by the ad hoc layout is not filled by the user, it's contents will be moved into the next group header and footer.

If there is no group filled at all, the content of the designed group will be moved into the page header and page footer depending on whether the group has set the flag 'Repeat group header in each page'.

This feature can be used to create grouped chart reports by placing the chart in the group footer of the first group. If there is no group at all, the chart will be moved into the page header and display all datasets.

Please note: Summary fields and group name fields are not moved since they are bound to a certain group. They will be removed if their group was not set/filled by the user. As a further notice, charts and cross-tab of the group footer will be moved into the page header, since they cannot be placed in the page footer.

4.5 Summaries

To enable the usage of summaries for the user, simply put a summary field into any section except the 'Detail Area'. The settings of this summary field will have no effect since they are overwritten by the user's choices.

At runtime the summary fields will be automatically aligned to the columns they summarize. They'll keep their size and vertical offset as well as the formatting.

The section a summary field is placed into defines the grouping and running total type of this field.

- if placed in a group header or footer, the summary will be grouped

- if placed in the page footer, the summary will be a running total
- if placed in the report header or footer, the summary field will summarize all records

Placing a summary field in several sections will cause ad hoc reporting to add similar summary to all of sections but with different group and running total types.

As an example, placing a summary in the group and report footer will create a grouped summary and a total for all columns selected by the user.

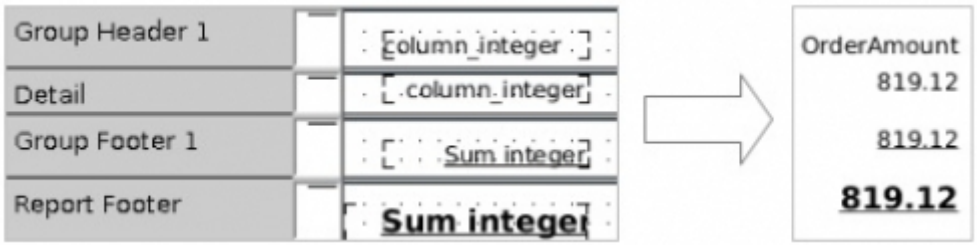

Figure 8: Summary Fields – layout and rendering result

Adding several summary fields to the same section in a real layout will have no effect. ad hoc reporting will only use the first summary field in a section as a template.

In a ready report, summary fields can only be modified by the user. Their position, number and appearance is fixed.

4.6 Formatting

The formatting of a field in an rendered report depends on three sources (in ascending priority):

- **Ad Hoc Layout:** In an ad hoc layout you are free to set any property of a field element as normal. The only limitation is the type of the field. Number properties for instance can only be set on number fields.
- **Ad Hoc Dataview:** Ad hoc dataviews are restricted to properties which are related to the type of data represented by a field. Pure styling properties cannot be set in an ad hoc dataview. So number properties are allowed for instance while the border or background color can only be set in the ad hoc layout. In general, the design of the rendered report will never be influenced by the chosen data source.

- **Field format set by the user:** The user is allowed to set the number or date/time format of any column. This setting has highest priority at it is set to any non-default value.

Property formulas are allowed in both the ad hoc layout and ad hoc dataviews. But keep in mind, that the type of the field at runtime may be different to the type of field used in the ad hoc layout. So, if you use 'currentFieldValue' in a property formula, the formula is only valid if the types match otherwise it will have no effect.

4.7 Chart and Crosstab

Ad hoc reporting supports to modify one chart and crosstab per ad hoc layout. If there are more charts or crosstabs in an ad hoc layout, they will not be modified. In a ready report the surplus charts / crosstabs will be rendered as they were designed. In a real layout they will be invisible since insufficiently configured charts and crosstabs are not displayed.

Every feature of charts and crosstabs can be used except the combined chart.

4.8 Special Fields

Every special field can be added to and ad hoc layout. The content of the following fields can be modified by the user:

- Title
- Author
- Comment

The following fields can be disabled by the user if present in the ad hoc layout:

- Print date
- Print time
- Page number
- Record Number

Please note, that special fields, which require information about the total number of pages of the rendered report will cause the whole report to be rendered before the first page is displayed. If these fields are not present the renderer will display the first page of a rendered report to the user as soon as it is available.

So, using special fields like 'Page number' may decrease the user experience of your ad hoc layout.

5 Ad Hoc Dataviews

An ad hoc dataview is a simplified representation of a complex database, which can be use as datasource for ad hoc reporting. It offers a simple list of columns that can be used in an ad hoc report instead of a large number of joined tables. End user don't have to know the structure of a database and can dive into ad hoc reporting.

An ad hoc dataview can be created using the i-net Designer. This enables you to use the same tools as required to design a normal report, like the visual linking wizard or the field property editors.

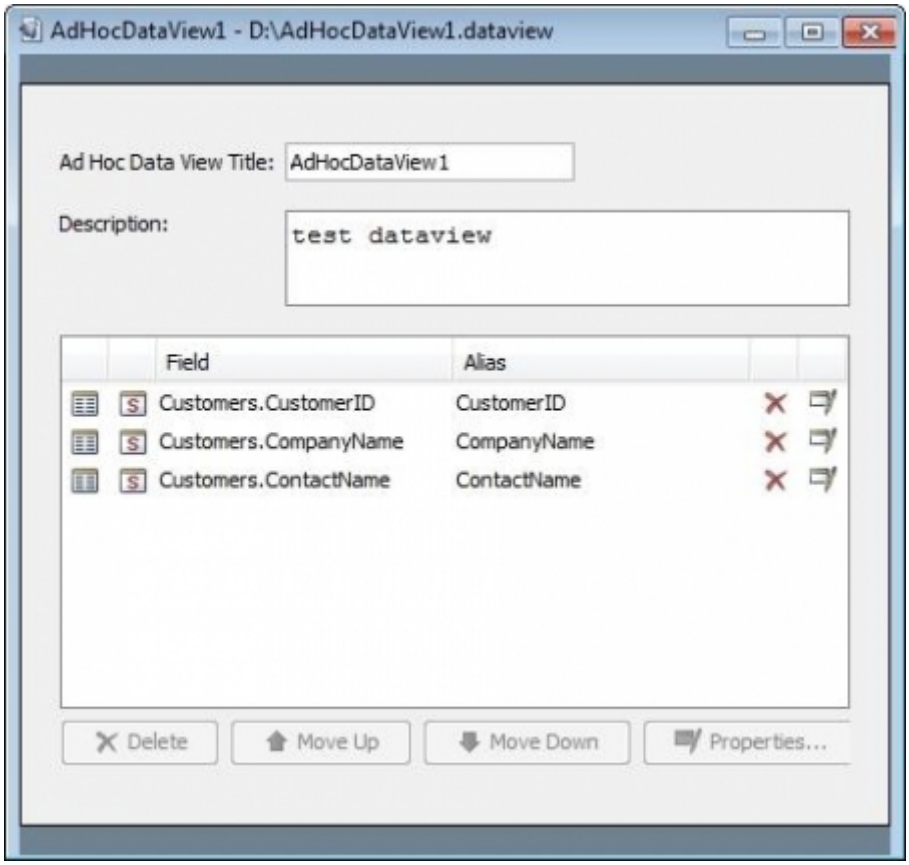

Figure 9: Ad Hoc Dataview editor

5.1 Database Fields

To provide access to certain database fields, use the visual linking wizard to add a datasource and it's tables to your report. Now you're able to add fields of these tables to your data view. Only fields which are added to the ad hoc dataview are visible to the user. The alias names and and field order is exactly what will be shown to the user. Please note, that duplicate names are not allowed as alias name.

5.2 Formula Fields

Formula fields can be used to create artificial fields. This is usually required to apply a complex formatting or to combine database fields into one field, for instance first and last name.

Once created a formula field can be added to the Dataview like a a database field. Formula fields do not have an additional alias. To change the display name of these fields, please change the name of the field itself.

The type of the field (database or formula) is hidden from the user, who will only see a plain list of field names and their data types.

5.3 Prompts

Ad hoc dataviews can use prompt fields as well. They can be defined and added to the ad hoc dataview like formula field. But their primary purpose is to allow the user to modify certain parameters for formula fields or the record selection formula.

5.4 Formating

The format of all fields in an ad hoc dataview can be changed. The available properties are limited to content formatting only. You cannot change the font style or colors here since these parameters are defined by the ad hoc layout.

If a property is set to a value other than default, this property will override the value defined by the ad hoc layout.

5.5 Record Selection

An additional feature of ad hoc dataviews is to define a filter by setting the record selection formula. This selection formula will be combined with the filter set by the user by an logical AND. The user will never see the additional filtering by the record selection formula.

5.6 Activation of Ad Hoc Dataviews

To activate ad hoc dataviews for a user, simply put the dataview files into the root of the ad hoc base directory or any of it's subdirectories. Since all ad hoc dataviews are presented as a plain list, it doesn't matter in which directory an ad hoc dataview is located.

If there is at least one ad hoc dataview found by ad hoc reporting, which can be utilized by the current user, this user is only allowed to use ad hoc dataviews instead of datasources and visual linking. Another option is to activate the CSV upload since an uploaded CSV file is presented as a ad hoc dataview.

The ad hoc dataviews visible to a user can be restricted by setting file permissions. It is highly recommended to review the file permissions for ad hoc dataviews if the public account is enabled.

6 Corporate Design

The ad hoc reporting feature of i-net Clear Reports supports the customization of the Remote Interface module.

6.1 Frameworks

The Remote Interface uses two different frameworks for the generation of the web GUI. Each framework is used for special purposes to achieve the best user experience.

The Remote Interface modules are created with the **Nextapp Echo Web Framework (Version 2)**. It ensures that only these components are loaded, which are needed for the active module.

The HTML Prompt Dialog is based on the Google Web Toolkit. Once the dialog is loaded, the user can configure the prompts without waiting for further content to be loaded.

6.2 Style definition for the Ad Hoc Reporting

The Echo Web Framework uses styles and stylesheets to configure the visual appearance of all components. The styles are defined in a special XML format, defined by the framework.

That results in the following limitations:

- It is not possible to change the design of the application with CSS. The Echo Web Framework has a defined set of properties that can be customized.
- Due to the possibility to create custom components and therefore the definition of custom properties, there is no detailed documentation on how to create the Stylesheet file and which tags and attributes can be used.

The following link shows a short example of a defined style for a button with the

style name "Button". In chapter Example a more complex example for the ad hoc reporting module is shown. In this example the colors and images of the module are replaced by another color scheme.

http://wiki.nextapp.com/echowiki/TipsAndTricks#tip7

If then whole appearance of the ad hoc reporting module shall be changed, the available properties can be found in the source code of the Echo Web Framework (version (2.1)), located at: http://echo.nextapp.com/site/echo2/download.

The Remote Interface defines a default set of styles for a unique visual appearance. Each single property can be replaced by overriding them with a new value. If a property, e.g. "background", shall be removed, it must be set to "null" with the following attribute:

```
<property name="background" setNull="true" />
```

If there is an error in the XML Stylesheet file, i-net Clear Reports will write the Exception into the log file at the time the Remote Interface is opened.

Errors that occur often:

- The specified resource does not exists: The ZIP file with the corporate design has to contain each image that is referenced in the stylesheet.
- No such attribute: The Echo Web Framework only accepts properties and attributes that are defined by the component. The source code of the framework contains the components class, e.g. Label.java and their peer LabelPeer.java, where all properties can be found.

Important: Please do not use the type "nextapp.echo2.app.ResourceImageReference" to reference images. The type "com.inet.remote.gui.StaticImageReference" should be used instead to ensure

that all images can be located by the i-net Clear Reports server.

6.2.1 Overview of the layout

To modify the design of the ad hoc web interface you'll need to know how the GUI is structured and which style name references which components on the screen.

The following image shows the main areas of the ad hoc remote interface.

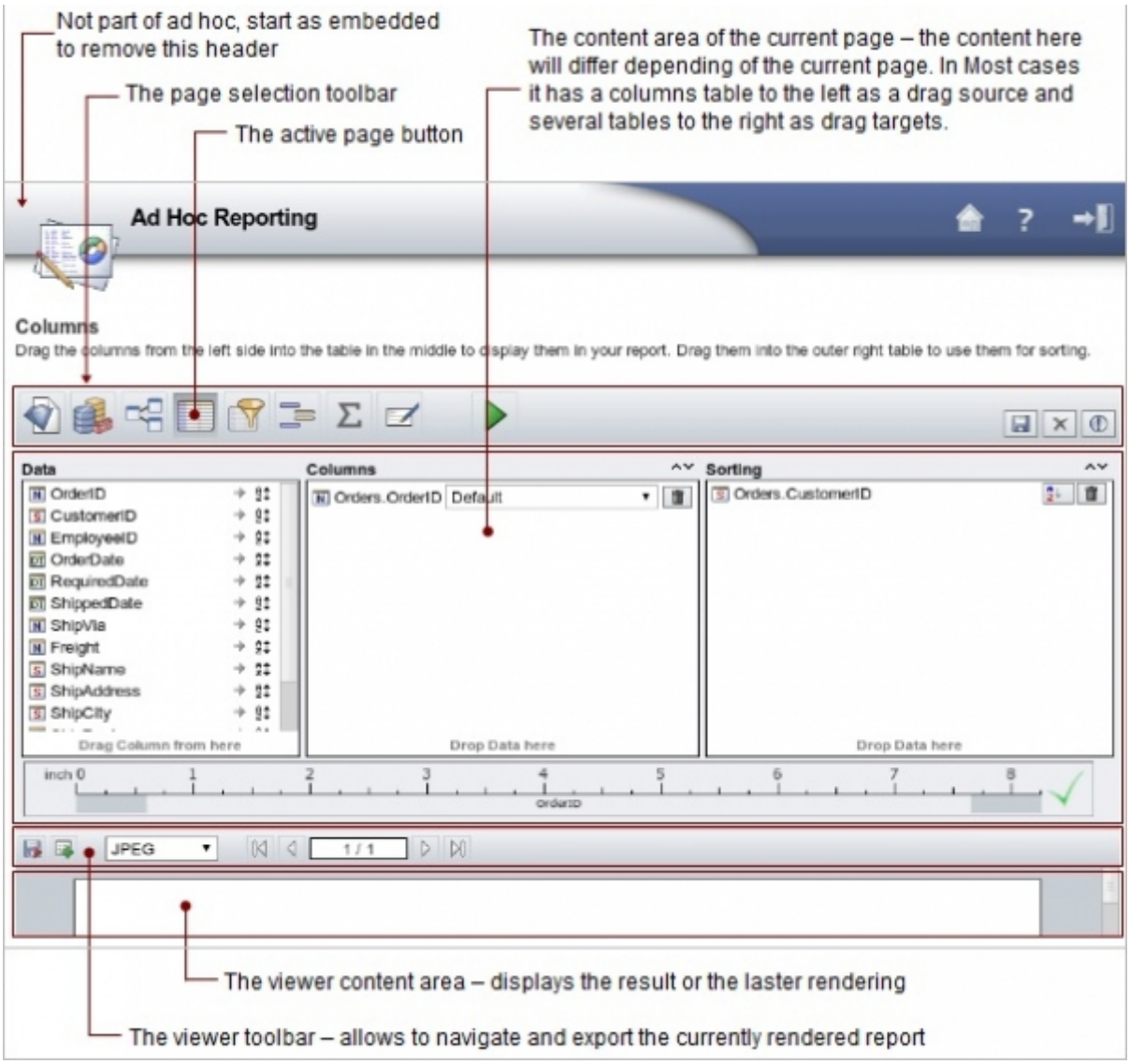

Figure 10: Basic web GUI components of ad hoc

An good entry point into changing the styles of the interface is to modify the background colors and images of the four areas shown in this image. This will set the base color scheme for the overall layout. The second step should be to set the appearance of the tables in the content area and the default buttons.

6.2.2 Available style names

The following table contains the available style names and component types that can be customized. A short description helps to find the component that shall be changed.

In order to set a style for all components of a special type, set an empty "name" attribute.

Style name	Type	Description
adhoc.button.columnSource	*nextapp.echo2.app.*Button	The button to add a column to a component (the drag&drop shortcut)
adhoc.button.chartSelect	*nextapp.echo2.app.*Button	The button to select a chart type
adhoc.button.chartSelect.selected	*nextapp.echo2.app.*Button	The button to select a chart type in selected state
adhoc.button.csvUpload	*nextapp.echo2.app.*Button	The CSV upload button on the dataview page
adhoc.button.export	*nextapp.echo2.app.*Button	(Viewer) Button for export of the current report
adhoc.button.first	*nextapp.echo2.app.*Button	(Viewer) Button, go to first page
adhoc.button.inTable	*nextapp.echo2.app.*Button	The default of buttons which are displayed as a table cell
adhoc.button.inTableHeader	*nextapp.echo2.app.*Button	The default of buttons which are displayed as a table header
adhoc.button.last	*nextapp.echo2.app.*Button	(Viewer) Button, go to last page

Style name	Type	Description
adhoc.button.next	*nextapp.echo2.app.*Button	(Viewer) Button, go to next page
adhoc.button.pictogram	*nextapp.echo2.app.*Button	Report page, the report preview&selection button
adhoc.button.pictogram.selected	*nextapp.echo2.app.*Button	Report page, the report preview & selection button, selected state
adhoc.button.prev	*nextapp.echo2.app.*Button	(Viewer) Button, go to previous page
adhoc.button.small	*nextapp.echo2.app.*Button	(Viewer) Toolbar default buttons style
adhoc.button.snapshot	*nextapp.echo2.app.*Button	(Viewer) Button to create a snapshot
adhoc.button.sort	*nextapp.echo2.app.*Button	Base style for the sort buttons displayed to the upper right of each table
adhoc.button.sort.down	*nextapp.echo2.app.*Button	Style of the "move entry downwards" button
adhoc.button.sort.up	*nextapp.echo2.app.*Button	Style of the "move entry upwards" button
adhoc.toolbar.button	*nextapp.echo2.app.*Button	Button, page selection toolbar
adhoc.toolbar.button.active	*nextapp.echo2.app.*Button	Button, page selection toolbar, if this page is currently active
adhoc.button.toolbarDefault	*nextapp.echo2.app.*Button	(Viewer) Toolbar default button style
adhoc.button.verify	*nextapp.echo2.app.*Button	Ruler, the button to verify the current columns
adhoc.button.warning	*nextapp.echo2.app.*Button	Viewer, the button which displays the render-warning
adhoc.checkbox	*nextapp.echo2.app.*CheckBox	Base style for all checkboxes
adhoc.column.pagecount	*nextapp.echo2.app.*Column	Viewer, page count box
adhoc.column.pagecount.inactive	*nextapp.echo2.app.*Column	Viewer, page count box, inactive
adHoc.default.contentBackground	*nextapp.echo2.app.*Component	The default background for all tables and lists

Style name	Type	Description
adHoc.default.contrastBackground	*nextapp.echo2.app.*Component	The background to contrast the default, e.g. used on the reports page
adhoc.dropDown.default	*echopointng.*MenuItem	Default for menu items in drop downs and the visual linking tables
adhoc.dropDown.default.button	*echopointng.*Menu	Default for buttons which open a drop down menu
adhoc.dropDown.default.button	*echopointng.*MenuButton	Default for buttons which open a drop down menu
adhoc.select.default	*nextapp.echo2.app.*SelectField	Default for all select fields (native drop downs)
adhoc.dropDown.chartSelect	*echopointng.*DropDown	The button to open the chart selection drop down
adhoc.dropDown.chartSelect.active	*nextapp.echo2.app.*DropDown	The button to open the chart selection drop down, active
adhoc.panel.error	*nextapp.echo2.app.*ContentPane	The background of the ruler if verify fails
adhoc.export.group	*nextapp.echo2.app.*Label	The group label line of the export dialog
adhoc.export.group	*echopointng.*LabelEx	The group label line of the export dialog
adhoc.default.foreground.active	*nextapp.echo2.app.*Component	Default foreground(font) color for active components
adhoc.default.foreground.inactive	*nextapp.echo2.app.*Component	Default foreground(font) color for inactive components
adhoc.grid.pictograms	*nextapp.echo2.app.*Grid	Report page, pictogram selection grid
adhoc.grid.spacer	*nextapp.echo2.app.*Grid	Report page, spacer to fill up the first row of pictograms
adhoc.grid.tablewrapper	*nextapp.echo2.app.*Grid	The wrapper box for the tables on the chart and crosstab page

Style name	Type	Description
adhoc.grid.tablewrapper.disabled	*nextapp.echo2.app.*Grid	The wrapper box for the tables on the chart and crosstab page, state disabled
adhoc.label.hint	*nextapp.echo2.app.*Label	The hint label which can be displayed instead of the columns list
adhoc.label.hint.small	*nextapp.echo2.app.*Label	The hint label to the bottom of the tables
adhoc.label.hint.small.error	*nextapp.echo2.app.*Label	The hint label to the bottom of the tables in case of an error
adhoc.label.pageBG	*nextapp.echo2.app.*Label	Styles for labels displayed on page background color
adhoc.label.pageBG.bold	*nextapp.echo2.app.*Label	Styles for labels displayed on page background color, bold
adhoc.label.pageBG.H1	*nextapp.echo2.app.*Label	Styles for labels displayed on page background color, H1
adhoc.chart.selectMenu	*nextapp.echo2.app.*SelectField	The select field to select the chart type
adHoc.default.pageBackground	*nextapp.echo2.app.*Component	The default component for the page background
adhoc.panel.page	*nextapp.echo2.app.*ContentPane	The base panel of all pages
adhoc.pane.ruler.inch	*nextapp.echo2.app.*ContentPane	The ruler image in inch scale
adhoc.pane.ruler.mm	*nextapp.echo2.app.*ContentPane	The ruler image in mm scale
adhoc.panel.reportSelect	*nextapp.echo2.app.*ContentPane	The report preview & selection panel
adhoc.panel.rulerMarker	*nextapp.echo2.app.*ContentPane	The background of the ruler markers
adhoc.ruler.units	*nextapp.echo2.app.*Label	The label which displays the mm or inch label
adhoc.split	*nextapp.echo2.app.*SplitPane	The default split used by Ad Hoc with default background and 5px separator

Style name	Type	Description
adhoc.split.header	*nextapp.echo2.app.*SplitPane	The default split used by Ad Hoc in white with default background and 5px separator
adhoc.split.reportSelect	*nextapp.echo2.app.*SplitPane	Split on the Report page between the path list and the previews
adhoc.split.rulerMarker.image	*nextapp.echo2.app.*SplitPane	Ruler, Marker split with marker image
adhoc.split.rulerMarker.noimage	*nextapp.echo2.app.*SplitPane	Ruler, Marker split without marker image
adhoc.split.viewer	*nextapp.echo2.app.*SplitPane	Split between the wizard and the viewer
adhoc.table.chartCrosstab	*nextapp.echo2.app.*Table	The data tables of the chart and crosstab page
adhoc.table.chartCrosstab	*echopointng.*TableEx	The data tables of the chart and crosstab page
adhoc.table.list	*nextapp.echo2.app.*Table	Style for tables which appear like a list
adhoc.table.rollover	*nextapp.echo2.app.*Table	The default for all tables which should use the rollover effect
adhoc.table.rollover	*echopointng.*TableEx	The default for all tables which should use the rollover effect
adhoc.default.table	*nextapp.echo2.app.*Table	The default for all tables
adhoc.table.upload	*echopointng.*TableEx	The columns tables of the CSV upload dialog
adhoc.table.wrapper.base	*nextapp.echo2.app.*ContentPane	Table wrapper, default insets
adhoc.table.wrapper.header	*nextapp.echo2.app.*Label	Table wrapper, header label
adhoc.table.wrapper.inner	*nextapp.echo2.app.*ContentPane	Table wrapper, inner content pane
adhoc.table.wrapper.inner.disabled	*nextapp.echo2.app.*ContentPane	Table wrapper, inner content pane, disabled
adhoc.table.wrapper.outer	*nextapp.echo2.app.*ContentPane	Table wrapper, border background and size, don't set borders here, set insets and bg-color

Style name	Type	Description
adhoc.table.wrapper.split	*nextapp.echo2.app.*SplitPane	The default split used by Ad Hoc in white with default background and 5px separator
adhoc.textarea.default	*nextapp.echo2.app.*TextArea	The default for all textarea components
adhoc.textfield.default	*nextapp.echo2.app.*TextField	The default for all textfield components
adhoc.toolbar.bgImage	*nextapp.echo2.app.*ContentPane	Toolbar background of the page selection
adhoc.toolbar.bgImageSmall	*nextapp.echo2.app.*ContentPane	Toolbar background of the viewer toolbar
adhoc.toolbar.helperRow	*nextapp.echo2.app.*Row	The small row to the right of the page selection toolbar
adhoc.toolbar.helperSplit	*nextapp.echo2.app.*SplitPane	Split between page buttons and help buttons
adhoc.panel.viewer	*nextapp.echo2.app.*ContentPane	The viewer content area
adhoc.window.dialog	*nextapp.echo2.app.*WindowPane	Default for all dialog windows. Can be used to change the dialog borders as well
adhoc.window.visualLinking	*nextapp.echo2.app.*WindowPane	The table windows of the visual linking page
adhoc.panel.wizard	*nextapp.echo2.app.*ContentPane	The overall background panel of ad hoc

6.2.3 Example

The following example shows how to manipulate some styles (in this case colors and background images) to get a fast visual change of the ad hoc reporting module.

```
<?xml version="1.0" encoding="UTF-8"?>
<stylesheet>
 <!-- A toolbar with a blue background -->
 <style name="adhoc.toolbar.bgImage"
```

```
type="nextapp.echo2.app.ContentPane">
  <properties>
   <property name="background" value="#123456" />
   <property name="backgroundImage" setNull="true" />
  </properties>
 </style>
 <!-- The viewer toolbar also in blue -->
 <style name="adhoc.toolbar.bgImageSmall"
type="nextapp.echo2.app.ContentPane">
  <properties>
   <property name="background" value="#123456" />
   <property name="backgroundImage" setNull="true" />
  </properties>
 </style>

 <!-- The toolbar buttons get a blue border (rolloverBorder and
pressedBorder stay unchanged) -->
 <style     name="adhoc.button.toolbarDefault"
    type="nextapp.echo2.app.Button">
  <properties>
   <property name="border">
    <border style="solid" size="1px" color="#123456" />
   </property>
  </properties>
 </style>
 <!-- The activated toolbar button gets a lighter background color -->
 <style     name="adhoc.toolbar.button.active"
    type="nextapp.echo2.app.Button">
  <properties>
   <property name="background" value="#dce6f1" />
   <property name="backgroundImage" setNull="true" />
  </properties>
 </style>


 <!-- Define the background colors for the layers in the pages -->
 <style     name="adHoc.default.contrastBackground"
    type="nextapp.echo2.app.Component">
  <properties>
```

```
   <property name="background" value="#becddc" />
  </properties>
 </style>
 <style     name="adHoc.default.pageBackground"
    type="nextapp.echo2.app.Component">
  <properties>
   <property name="background" value="#dce6f1" />
  </properties>
 </style>
 <style     name="adHoc.default.contentBackground"
    type="nextapp.echo2.app.Component">
  <properties>
   <property name="background" value="#eff4f9" />
  </properties>
 </style>

 <!--    The separators between the tables shall have the same color
like the
   page background -->
 <style name="adhoc.split" type="nextapp.echo2.app.SplitPane">
  <properties>
   <property name="separatorColor" value="#dce6f1" />
  </properties>
 </style>

 <!--    All buttons (except the toolbar buttons) with bold font, blue
   background and image and white foreground. The rollover changes the
   background image -->
 <style name="" type="nextapp.echo2.app.Button">
  <properties>
   <property name="font">
    <font bold="true" />
   </property>
   <property name="foreground" value="#ffffff" />
   <property name="rolloverForeground" value="#ffffff" />
   <property name="pressedForeground" value="#ffffff" />
   <property name="disabledForeground" value="#999999" />

   <property name="background" value="#123456" />
```

```
  <property name="rolloverBackground" value="#123456" />
  <property name="pressedBackground" value="#123456" />
  <property name="disabledBackground" value="#123456" />

  <property name="border">
   <border style="solid" size="1px" color="#dce6f1" />
  </property>
  <property name="rolloverBorder">
   <border style="solid" size="1px" color="#eff4f9" />
  </property>
  <property name="backgroundImage">
   <fill-image>
    <image type="com.inet.remote.gui.StaticImageReference">
     <resource-image-reference
       resource="example/buttonSmall.png" />
    </image>
   </fill-image>
  </property>
  <property name="rolloverBackgroundImage">
   <fill-image>
    <image type="com.inet.remote.gui.StaticImageReference">
     <resource-image-reference
       resource="example/buttonSmallRollover.png" />
    </image>
   </fill-image>
  </property>
  <property name="pressedBackgroundImage">
   <fill-image>
    <image type="com.inet.remote.gui.StaticImageReference">
     <resource-image-reference
       resource="example/buttonSmallPressed.png" />
    </image>
   </fill-image>
  </property>
 </properties>
</style>

<!-- The report layout selection gets blue borders -->
<style name="adhoc.button.pictogram" type="nextapp.echo2.app.Button">
```

```
 <properties>
  <property name="border">
   <border style="solid" size="2px" color="#becddc" />
  </property>
  <property name="rolloverBorder">
   <border style="solid" size="2px" color="#123465" />
  </property>
  <property name="pressedBorder">
   <border style="solid" size="2px" color="#000000" />
  </property>
 </properties>
</style>
<!-- The selected report layout has an orange border -->
<style    name="adhoc.button.pictogram.selected"
   type="nextapp.echo2.app.Button">
 <properties>
  <property name="border">
   <border style="solid" size="2px" color="#f9b41e" />
  </property>
  <property name="rolloverBorder">
   <border style="solid" size="2px" color="#123465" />
  </property>
  <property name="pressedBorder">
   <border style="solid" size="2px" color="#000000" />
  </property>
 </properties>
</style>

<!-- The selection color in tables with blue and orange -->
<style name="adhoc.default.table" type="nextapp.echo2.app.Table">
 <properties>
  <property name="rolloverForeground" value="#ffffff" />
  <property name="rolloverBackground" value="#123465" />
  <property name="selectionForeground" value="#000000" />
  <property name="selectionBackground" value="#f9b41e" />
 </properties>
</style>

<!--    The DropDown menus and the buttons in the tables also with the
```

```
blue
   and orange selection -->
 <style name="adhoc.dropDown.default" type="echopointng.MenuItem">
  <properties>
   <property name="rolloverEnabled" value="true" />
   <property name="pressedEnabled" value="true" />
   <property name="foreground" value="#000000" />
   <property name="rolloverForeground" value="#ffffff" />
   <property name="pressedForeground" value="#666666" />
   <property name="disabledForeground" value="#AAAAAA" />
   <property name="rolloverBackground" value="#123465" />
   <property name="pressedBackground" value="#f9b41e" />
   <property name="disabledBackground" value="#ffffff" />
  </properties>
 </style>
 <style name="adhoc.button.inTable" type="nextapp.echo2.app.Button">
  <properties>
   <property name="border">
    <border style="solid" size="1px" color="#becddc" />
   </property>
   <property name="rolloverBorder">
    <border style="solid" size="1px" color="#123465" />
   </property>
  </properties>
 </style>
 <!-- The border of the dialogs are generated out of 8 graphics -->
 <style name="" type="nextapp.echo2.app.WindowPane">
  <properties>
   <property name="titleBackground" setNull="true" />
   <property name="titleForeground" value="#123465" />
   <property name="titleBackgroundImage">
    <fill-image>
     <image type="com.inet.remote.gui.StaticImageReference">
      <resource-image-reference
        resource="example/window_title.png" />
     </image>
    </fill-image>
   </property>
   <property name="border">
```

```
<fill-image-border    content-insets="18px 18px 18px 18px"
     border-insets="18px 18px 18px 18px">
 <border-part position="top-left">
  <fill-image>
   <image type="com.inet.remote.gui.StaticImageReference">
    <resource-image-reference
     resource="example/window_tl.png" />
   </image>
  </fill-image>
 </border-part>
 <border-part position="top">
  <fill-image>
   <image type="com.inet.remote.gui.StaticImageReference">
    <resource-image-reference
     resource="example/window_t.png" />
   </image>
  </fill-image>
 </border-part>
 <border-part position="top-right">
  <fill-image>
   <image type="com.inet.remote.gui.StaticImageReference">
    <resource-image-reference
     resource="example/window_tr.png" />
   </image>
  </fill-image>
 </border-part>
 <border-part position="right">
  <fill-image>
   <image type="com.inet.remote.gui.StaticImageReference">
    <resource-image-reference
     resource="example/window_r.png" />
   </image>
  </fill-image>
 </border-part>
 <border-part position="bottom-right">
  <fill-image>
   <image type="com.inet.remote.gui.StaticImageReference">
    <resource-image-reference
     resource="example/window_br.png" />
```

```
      </image>
     </fill-image>
    </border-part>
    <border-part position="bottom">
     <fill-image>
      <image type="com.inet.remote.gui.StaticImageReference">
       <resource-image-reference
        resource="example/window_b.png" />
      </image>
     </fill-image>
    </border-part>
    <border-part position="bottom-left">
     <fill-image>
      <image type="com.inet.remote.gui.StaticImageReference">
       <resource-image-reference
        resource="example/window_bl.png" />
      </image>
     </fill-image>
    </border-part>
    <border-part position="left">
     <fill-image>
      <image type="com.inet.remote.gui.StaticImageReference">
       <resource-image-reference
        resource="example/window_l.png" />
      </image>
     </fill-image>
    </border-part>
   </fill-image-border>
  </property>
 </properties>
</style>
<!--    The table windows in the Visual Linking Page shall have a blue
  window frame which is 2px wide -->
<style    name="adhoc.window.visualLinking"
   type="nextapp.echo2.app.WindowPane">
 <properties>
  <property name="titleBackground" value="#123456" />
  <property name="titleForeground" value="#ffffff" />
  <property name="titleFont">
```

```
    <font bold="true" size="10pt" />
   </property>
   <property name="titleBackgroundImage" setNull="true" />
   <property name="border">
    <fill-image-border content-insets="2px 2px 2px 2px"
        border-insets="2px 2px 2px 2px" color="#123456">
    </fill-image-border>
   </property>
  </properties>
 </style>
</stylesheet>
```

6.3 Style definition for the HTML Prompt Dialog

The prompt dialog is shown when a data view contains parameter fields that are required. To customize the colors of this prompt dialog, you'll need to customize the CSS file "colors.css" which can be found in the "client/prompt" folder of your installation.

There are three parts to this CSS, divided by comments. The colors in the first part are intended to be the primary color in three various shades from dark to light. The second part contains the secondary background color, and the third contains the borders' colors.

6.4 Deploying the Corporate Design

The easiest way to deploy a corporate design is to create a ZIP file that contains the stylesheet XML file in the root (no sub folders) and the images (in the folders that are referenced in the XML file).

Example of the structure:

- remoteInterfaceStyles.xml
- my_images
- toolbar_bg.png
- button_bg.png

- ...

Now the ZIP file can be copied into the lib folder of the i-net Clear Reports installation. With the next restart of the server, the ZIP file is loaded automatically.

6.5 Starting Ad Hoc Reporting with the new design

After deploying the corporate design ZIP file and restarting the i-net Clear Reports server, the template can be chosen.

If the stylesheet file is named "remoteInterfaceStyles.xml", it is applied automatically.

If the file is named differently, the template can be set using the "template" URL parameter. For example if the file is named "myDesign.xml", the following URL opens the ad hoc reporting module wit the specified template:

```
  http://host/context/adhoc/?template=myDesign.xml
or
  http://host/context/adhoc/?template=myDesign
```

Setting a template that does not exist, will result in the style that is provided by i-net Clear Reports.

7 Typical operating scenarios

There are two implementations of the ad hoc reporting client available

- Java Swing based client, which can be used as a standalone client or embedded in Java applications
- Echo web application framework based client, requires only a web browser on the client operating system to run

This chapter describes the most common usages of the ad hoc reporting clients.

7.1 Remote Interface Module

Using ad hoc reporting as a module in the remote interface of i-net Clear Reports is one of it's primary modes of operation. Simply enable the remote interface and the ad hoc reporting in the configuration manager of your server. For further details how to set up the configuration have a look at chapter Configuration.

7.2 Embedding the Ad Hoc Reporting Module

The ad hoc reporting module of the Remote Interface can be embedded into an existing website. The embedded mode hides the global toolbar of the Remote Interface to ensure that only the module content is visible.

Due to the necessary HTML content, an iframe must be used. The following code is an example for embedding the module that runs on a server "host" with the application context "context".

```
<iframe src="http://host/context/adhoc/?embedded=true" width="800"
height="600"></iframe>
```

If authentication is activated at the i-net Clear Reports server, it is possible to use the "Guest Account" feature. This will log in any user with the configured account

name. So the user does not need to enter a user name and password but is logged in with an account that's rights can be administrated.

The following additional URL parameters can be used to modify the appearance of the ad hoc reporting:

- **showhel=true** - **- Expands the help panel to the right of the ad hoc panel on start up. This is useful if the ad hoc reporting is primarily used by less experienced us** * showdescription=false **-lse** - Hides the page title and description area. This will save 50 PX of the ad hoc frame height.

7.3 Java Applet

The Swing client of ad hoc reporting can be run as an applet as well. This applet will be delivered automatically by your i-net Clear Reports server, if the location <server-URL>/adhoc is requested by a browser.

To embed the ad hoc applet in your website simply insert the following applet code:

```
<applet code="com.inet.adhoc.client.loader.AHLoader" codebase="../core"
archive="adhocLoader.jar">
          <param name="codebase_lookup" value="false">
</applet>
```

7.4 Embedded Java Application

The Swing client of ad hoc reporting is based on a JPanel which can easily be embedded into any Java Swing application.

You can choose to either use the client with report viewer, which is represented by the class com.inet.adhoc.client.AHClient or use only the pure wizard component, which is com.inet.adhoc.client.AHWizard.

7.4.1 Client with Viewer

Using the client with report viewer is an "out of the box" solution with one limitation: the class AHClient has no function to return the generated Engine or report programmatically. It's intended to only serve as report generator for the user.

To use the AHClient, simply instantiate the class with the URL to you server and add it to the component tree of your Java Swing application.

```
AHClient client = new AHClient( serverURL, ahHelpProvider );
```

The parameter AHHelpProvider can be set to null. In that case the client will try to resolve the help pages by the serverURL and open the pages in the default browser.

7.4.2 Wizard

The AHWizard only contains the ad hoc report generation pages. It does not contain a render component like the Swing report viewer of the AHClient. On the other hand, it's more customizable since you can modify it's configuration and set custom request handlers to connect to multiple servers at once.

Since the request handlers can be accessed, you've got the ability to request the generated engine from the server by sending the following request.

```
IResponse response = requestHandler.handleRequest( request, false );
XMLSerializableByteArray engineData =
(XMLSerializableByteArray)response.getProperties().get(
  AHConstants.KEY_ENGINE_DATA );
if( engineData != null ){
 return RDC.loadEngine(
   new URL( "file:/"),
   new ByteArrayInputStream( engineData.getValue() ),
```

```
    Engine.EXPORT_DATA, null );
}
```

This type of embedding is also used for the i-net Designer.