Панель дополнительных инструментов для разработчика на InterSystems IRIS

Панель дополнительных инструментов для мониторинга и исследования ошибок приложений и интеграционных решений на платформе данных InterSystems IRIS, интеграционной платформе Ensemble и СУБД Caché, или история еще одного велосипеда.

В этой статье я хочу рассказать о приложении, которым, наряду со стандартными средствами администрирования, пользуюсь ежедневно при мониторинге приложений и интеграционных решений на платформе InterSystems IRIS и нахождении ошибок при их возникновении.
Решение включает просмотр и редактирование глобальных массивов, выполнение запросов (включая JDBC / ODBC), отправка результатов поиска по электронной почте в виде архивированных XLS-файлов. Просмотр объектов классов с возможностью редактирования. Несколько простых графиков по протоколам системы.

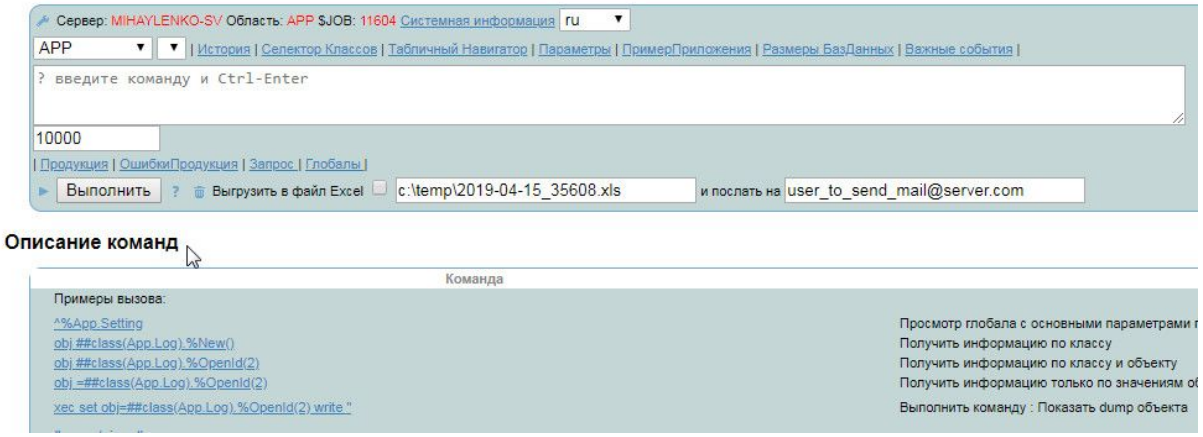
Это CSP-приложение, на основе jQuery-UI, chart.js, jsgrid.js
Если интересно, то прошу под кат и в репозиторий.

Все началось с изучения вопроса, как логировать изменения объектов в InterSystems IRIS, Ensemble и СУБД Caché.

Прочитав прекрасную статью об этом, я форкнул проект. и начал его допиливать для своих нужд.

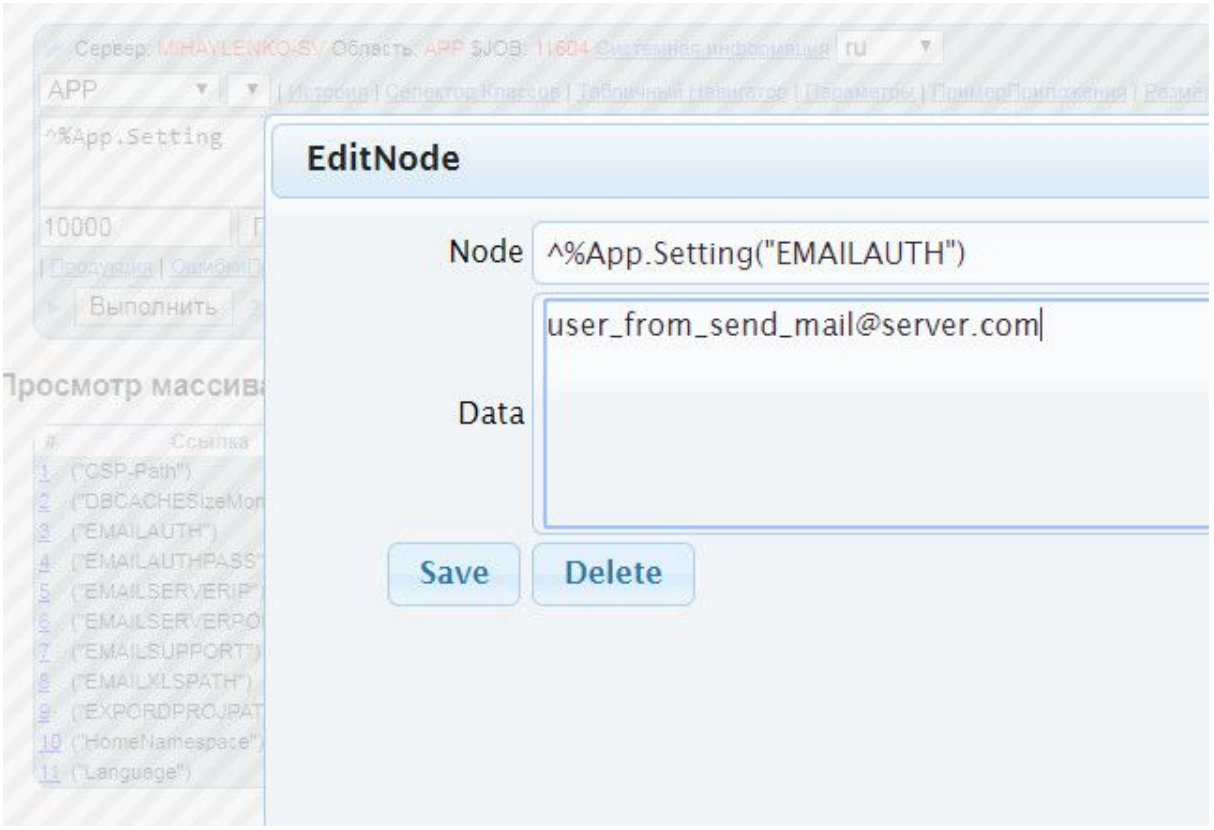
В результате получилось решение, которое реализовано как панель подкласса %CSP.Util.Pane, в котором есть основное окно для команд и кнопка «Выполнить», плюс настройки уточнений для команд.

При вводе “?” получаем краткое описание этих команд:



Глобалы

Самая частая моя команда — просмотр глобала. Как правило это глобал протокола при отладке своего или чужого проекта. Его можно посмотреть и в обратном порядке, а также наложив фильтр как на ссылку, так и на данные. Найденные узлы можно редактировать и удалять:



Можно удалить весь глобал введя в команде после имени минус ^logMSW-
Но удалить так можно только глобалы начинающиеся на ^log (протокольные глобалы), т.е. реализовано ограничение от случайного удаления.

Если после имени вводить “*” то получим список глобалов с дополнительными характеристиками. Вторая “*” — добавит новое поле “Allocated MB”, а еще одна звездочка — “Used MB” Это объединение двух отчетов и разделение на “звездочки” сделано, чтобы разделить часто долго формируемый отчет по занятым блокам больших глобалов.

Из этой таблички можно по активным ссылкам перейти к просмотру самого глобала или к его просмотру/редактированию стандартным способом из портала управления кликнув в поле Permission на R или W.

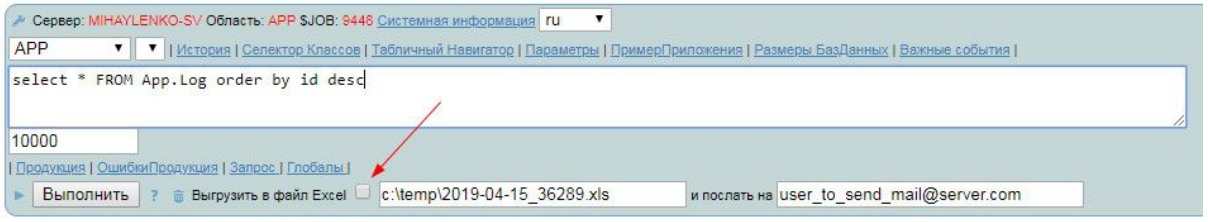
Запросы

Конвертирование отчета в формат Excel

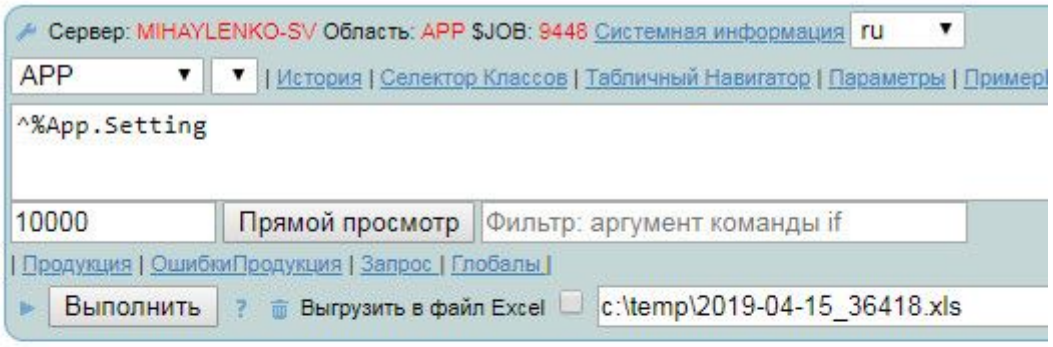
Вторая функция, по частоте использования, это выполнение запросов. Для этого sql-утверждение вводим в качестве команды.

Основное, чего мне хватало в стандартном Портале управления системы, это выполнение запросов по настроенным в СУБД JDBC-/ODBC-источникам и вывод результатов в формате XLS, архивирование и посылка файла на почту. Для этого в моем инструменте перед выполнением команды нужно включить чек-бокс “Выгрузить в файл Excel”.

Эта возможность экономит мне много времени в повседневной текучке, а готовые модули я успешно встраиваю в новые приложения и интеграционные решения.



Но для этого предварительно нужно настроить путь создания файлов на сервере и учётные данные пользователя и почтового сервера, для этого в свою очередь нужно редактировать узлы глобала настройки программы ^%App.Setting.



Просмотр массива : ^%App.Setting в области APP

#	Ссылка	
1	("CSP-Path")	C:\InterSystems\Ensemble17\CSP\app
2	("DBCACHESizeMon")	CACHESYS,CACHEAUDIT
3	("EMAILAUTH")	user_from_send_mail@server.com
4	("EMAILAUTHPASS")	12345
5	("EMAILSERVERIP")	127.0.0.1
6	("EMAILSERVERPORT")	25
7	("EMAILSUPPORT")	user_to_send_mail@server.com
8	("EMAILXLSPATH")	c:\temp\
9	("EXPORDPROJPATH")	c:\temp\source\
10	("HomeNamespace")	APP
11	("Language")	ru

Сохранение отчетов в глобале

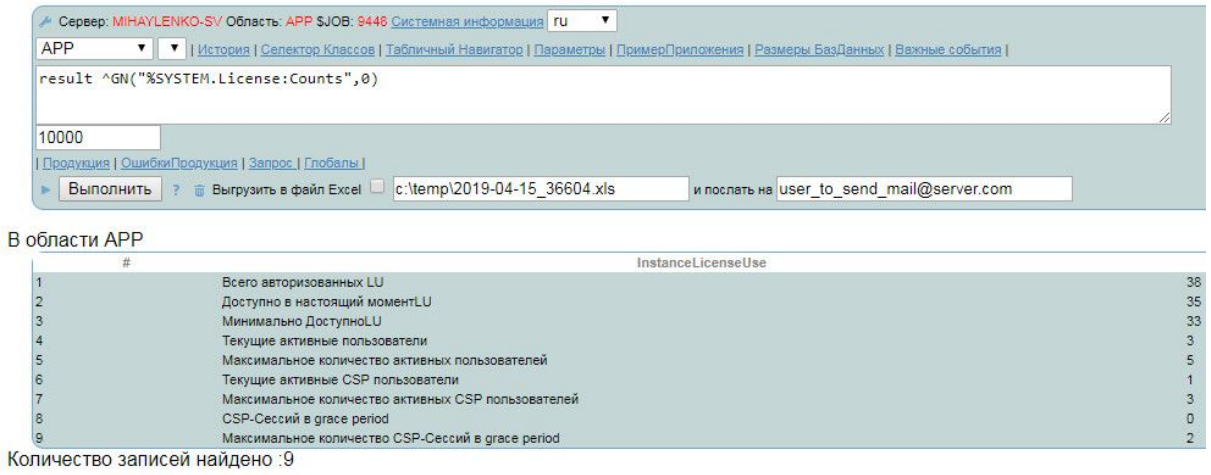
Очень часто требуется результаты выполнения отчета сохранить в глобале. Для этого я применяю процедуры:

Для JDBC:	##class(App.sys).SqlToDSN
Для ODBC:	##class(App.sys).SaveGateway
Для SQL выражений:	##class(App.sys).SaveSQL
Для Query:	##class(App.sys).SaveQuery

Например, если в панели командой

```
xec do ##class(App.sys).SaveQuery("%SYSTEM.License:Counts","^GN",0)
```

сохраним в массиве ^GN результат запроса подсчёта использования лицензий, а посмотреть, что сохранили в панели можно командой: `result ^GN("%SYSTEM.License:Counts",0)`



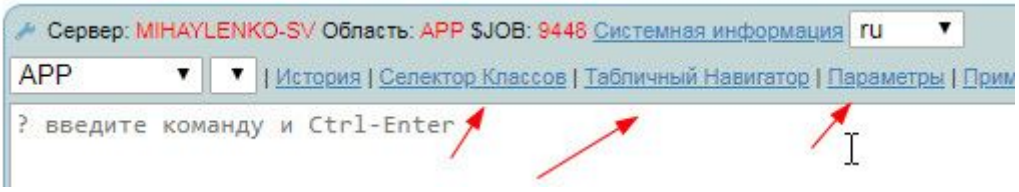
Модули дополненной функциональности

И второе улучшение, которое сильно упростило и автоматизировало мне работу — это реализация возможности выполнять специально написанные модули при формировании каждой строки запроса. Этим самым я могу на лету за один проход встраивать в отчет новый функционал, например, активные ссылки для дополнительных операций над данными.

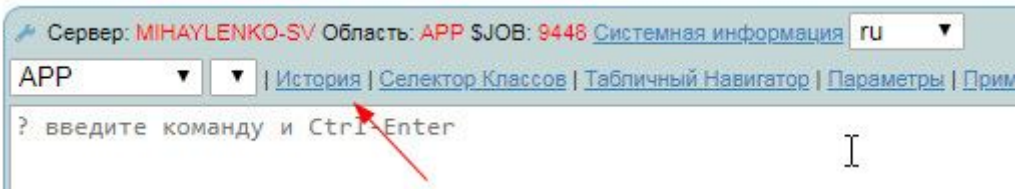
Пример 1: Работа с классом App.Parameter

Создать параметр через «Табличный навигатор»

Редактировать параметр через «Параметры»



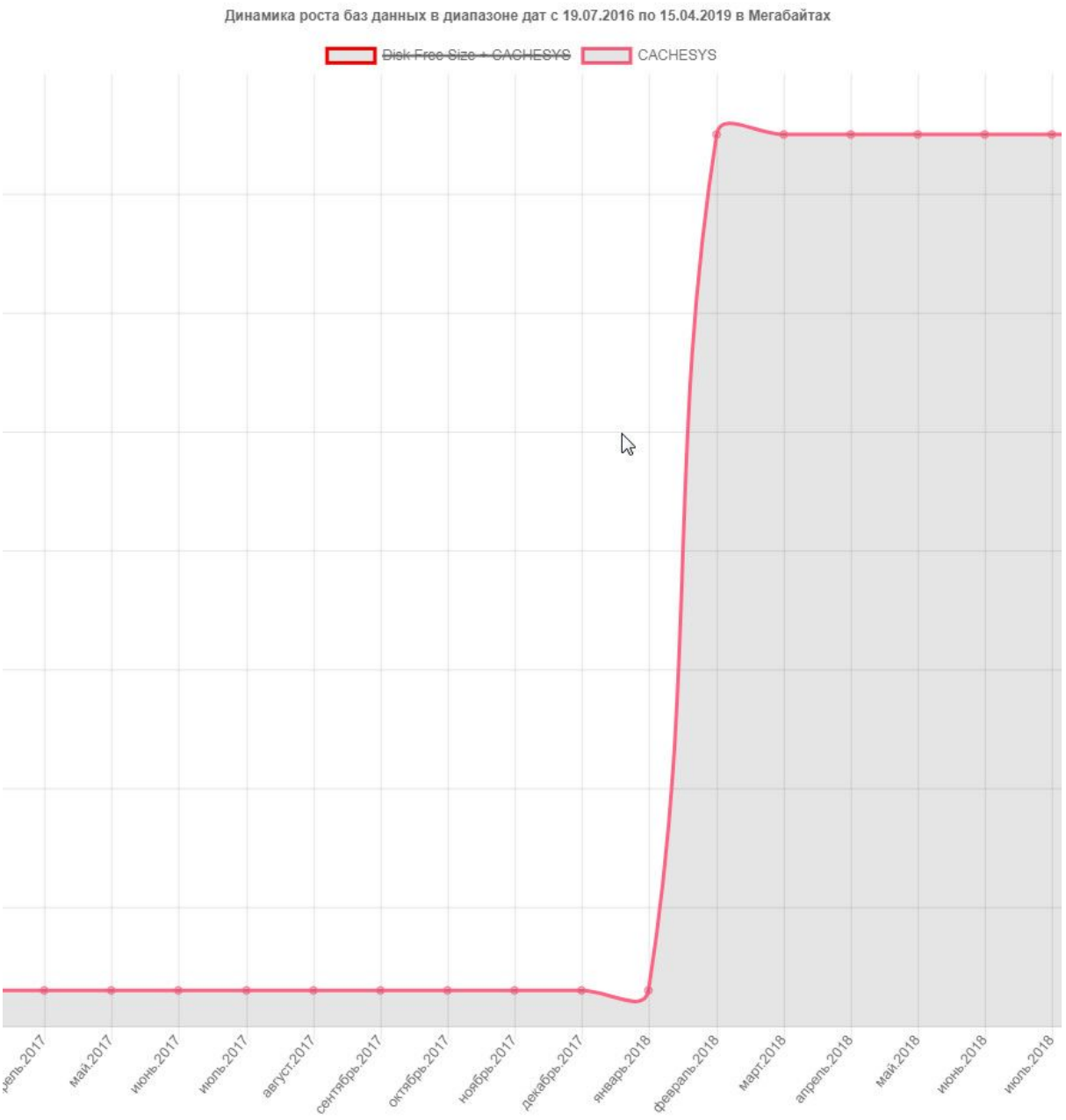
Пример 2: Просмотр глобала через ссылку «История»



Графики

Под впечатлением от статьи [9] и для визуализации роста баз данных создана страница, на которую выводится помесячный график размеров баз данных, созданный по файлу iris.log(cconsole.log) по записям “Expand” ретроспективно от текущего дня.

Для примера еще создан график событий в InterSystems IRIS, который тоже формируется по файлу протокола:



Ссылки на материалы:

[1] подсистема логирования в Каше
[2] Каша быстрого приготовления — делаем CRUD в Caché с помощью jqGrid
[3] Альтернативные SQL-менеджеры для СУБД Caché
[4] Примеры генерации и отправки Email средствами СУБД Caché
[5] Cache + jQuery. Быстрый старт
[6] Развертывание приложений
[7] UDL-поддержка
[8] Просмотр глобалов в Портале Управления СУБД Caché

[9] Prometheus с Caché
[10] Локализация в СУБД Caché

Благодарю авторов этих и других статей, которые помогли мне создать этот инструмент.

P.S. Этот проект развивается и многие задумки еще не реализованы. В ближайшее время планирую сделать:

1. Шаблон приложения на фрэймворке uikit
2. Автодокументирование кода форматом Doxegen с интеграцией в CStudio