jPALIO SA

Agile Technology. Peace of Mind.



Dokumentacja **jPALIO 7.4**

Spis treści

1. Analiza składni 1

1.1. Model zawartości obiektu 1

1.2. Inspekcje 2

1.3. Wyłączanie kompilacji i inspekcji 2

1.4. Podpowiadanie i uzupełnianie kodu 3

1.5. Podpowiadanie parametrów 3

2. Wersjonowanie 3

2.1. Tworzenie rewizji 4

2.2. Lista zmodyfikowanych obiektów 4

2.3. Porównanie rewizji 6

3. Okno porównywania obiektów 7

3.1. Markery podświetlające różnice 8

3.2. Porównywanie dwóch wersji archiwalnych 10

3.3. Mechanizm porównywania 10

4. Panel administracyjny 12

4.1. Dostęp 12

4.2. Autoryzacja 12

4.3. Zarządzanie serwerem 13

4.4. Zarządzanie instancją 15

5. Zarządzanie aktywnymi łatkami 16

5.1. Zarządzanie aktywnymi łatkami w module „Obiekty” 16

5.2. Zmiana nazwy modułu „Projekt” na „Historia” 17

6. Wgrywanie łatek 18

6.1. Zadkładka „Domyślne” 19

6.2. Zakładka „Zaawansowane” 19

6.3. Zakładka „Kompatybilne” 20

7. Automatyczna aktualizacja 20

7.1. Konfiguracja instancji 21

7.2. Tworzenie/aktualizacja struktur dla zewnętrznych aplikacji 21

7.3. Zasady działania AutoUpdate 21

8. Konfiguracja serwera 22

8.1. Konto administratora serwera 22

8.2. Wybór uruchamianej wersji jDESIGNER'a 22

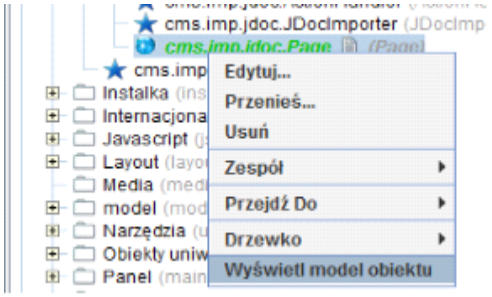
1. Analiza składni

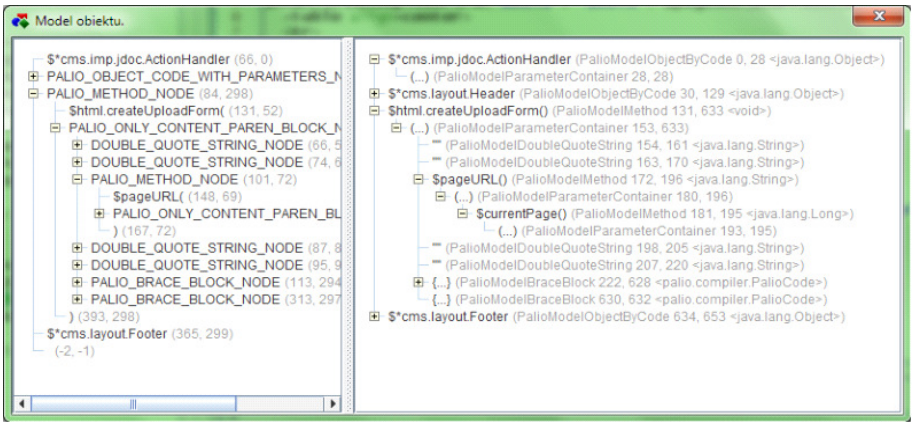
Od wersji jPALIO 7.4 jDESIGNER posiada wbudowany kompilator po stronie klienta. Pozwala on na wykrywanie prostych błędów i potencjalnych pomyłek programisty oraz ostrzeganie o nich bezpośrednio w kodzie.

Kompilacja i podświetlanie błędów odbywa się w locie zawsze po otwarciu nowego obiektu i po każdej zmianie zawartości. W obecnej wersji wspierane są obiekty typu jPALIO.

1.1. Model zawartości obiektu

Po wygenerowaniu przez kompilator modelu obiektu możliwy jest jego podgląd. W tym celu należy nacisnąć prawym przyciskiem na obiekt w drzewie obiektów i wybrać opcję wyświetl model obiektu.



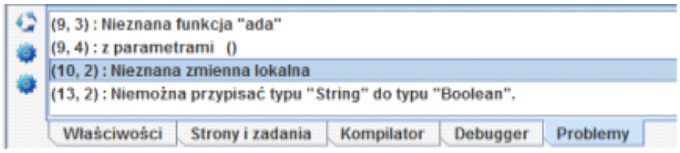


Błędy wykryte przez kompilator wyświetlane są w kodzie, poprzez zmianę stylu tekstu lub podkreślenie fragmentu tekstu w edytorze.

```
$ada( ) $// brak funkcji
$@ala $// brak zmiennej ala

$=(@ala, true)
$=(@ala, "String") $// niezgodność typów
```

Najechanie myszką na błąd w kodzie powoduje pojawienie się okienka wyświetlającego treść błędu. Dodatkowo poniżej edytora w zakładce problemy, widoczny jest spis wszystkich błędów wraz z pozycją gdzie dany błąd występuje. Kliknięcie myszką na błąd powoduje automatyczne przejście karetki do miejsca jego wystąpienia.



1.2. Inspekcje

Dodatkowo po utworzeniu modelu obiektu podczas kompilacji wykonywanych jest szereg inspekcji kodu. Mają one na celu wykrycie bardziej zaawansowanych błędów w kodzie.

Przykładowe inspekcje to:

- Wyszukiwanie nieistniejących elementów jPALIO (obiekty, strony, przywileje, itd.)
- Wyszukiwanie literówek w zmiennych globalnych
- Niepotrzebne mapowanie parametrów
- Zaznaczanie przestarzałych metod

```
$zmienna
$zmienan $//literówka

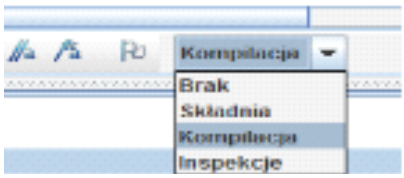
$*nieistnieje $//nieistniejący obiekt

$pageURL((object)3) $//przestarzała metoda oraz
                    $//niepotrzebne mapowanie
```

1.3. Wyłączanie kompilacji i inspekcji

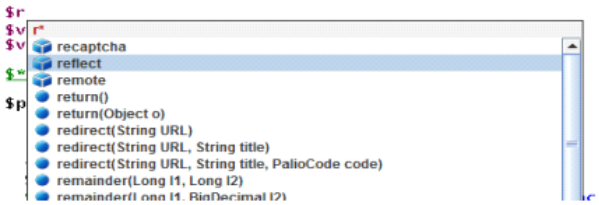
Wykonywanie inspekcji jest pracochłonne dla procesora i można je wyłączyć. Poniżej edytora znajduje się rozwijana lista, w której mamy możliwość wyboru jednej spośród czterech opcji opisanych poniżej:

- Brak – brak jakiegokolwiek kolorowania składni w edytorze
- Składnia – analizowana jest tylko składnia i na jej podstawie kolorowane są elementy tekstu
- Kompilacja – wykonywana jest analiza składni oraz kompilacja obiektu i na podstawie stworzonego modelu i składni następuje kolorowanie tekstu, wraz z zaznaczeniem podstawowych błędów
- Inspekcje – wykonywana jest analiza składni, kompilacja obiektu, oraz dodatkowo inspekcje. Na podstawie wszystkich tych informacji następuje kolorowanie tekstu, oraz zaznaczanie błędów.



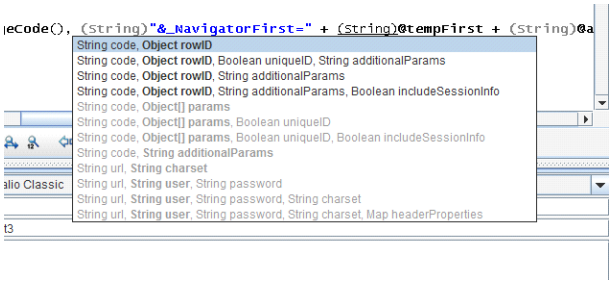
1.4. Podpowiadanie i uzupełnianie kodu

Obecnie istnieje już funkcja podpowiadania kodu po kliknięciu Ctrl + Spacja. Edytor może podpowiadać funkcje jPalio lub kod obiektu.



1.5. Podpowiadanie parametrów

W edytorze jest dostępna również funkcja podpowiadania parametrów. Po umieszczeniu karetki wewnątrz nawiasów metody i kliknięciu Ctrl + Shift + Spacja następuje podpowiadanie parametrów metody.



2. Wersjonowanie

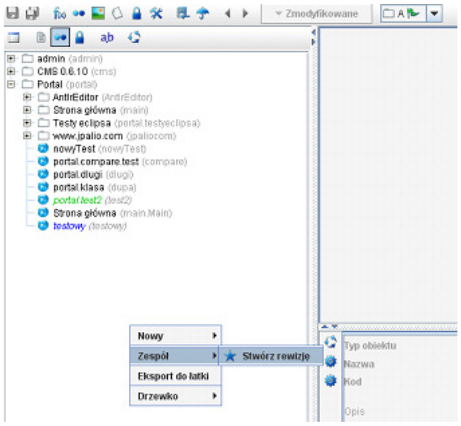
Wersjonowanie jest mechanizmem pozwalającym przechowywać w bazie historyczne wersje obiektów w celu późniejszego śledzenia historii zmian obiektu. Każdy użytkownik może w dowolnym momencie pracy podglądać zarchiwizowane treści obiektów oraz przywracać ich zawartość bądź fragmenty.

Wyróżnia się dwa rodzaje obiektów archiwalnych:

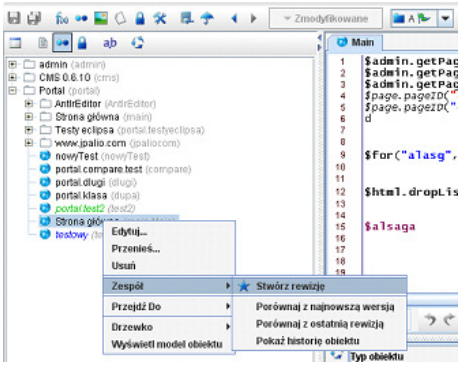
- Automatyczne archiwizacje – tworzone automatycznie w trakcie pracy z obiektem co 15 min od ostatniej archiwizacji a także przy próbie zapisu zmian w obiekcie przez innego użytkownika. Tworzona jest również przed usunięciem obiektu.
- Rewizja – tworzona na żądanie użytkownika przy pomocy okna tworzenia rewizji, opatrzona komentarzem oraz unikalnym numerem. Odpowiada rewizjom systemu kontroli wersji SVN.

2.1. Tworzenie rewizji

W celu wywołanie okna umożliwiającego stworzenie rewizji należy kliknąć prawym przyciskiem myszy na obszarze okna, zawierającego listę obiektów, a następnie wybrać z menu: „Zespół” -> „Stwórz rewizję”. W zależności od aktualnej pozycji kursora, możemy uzyskać następujące rezultaty:
Wybrano obszar nie należący do żadnego obiektu



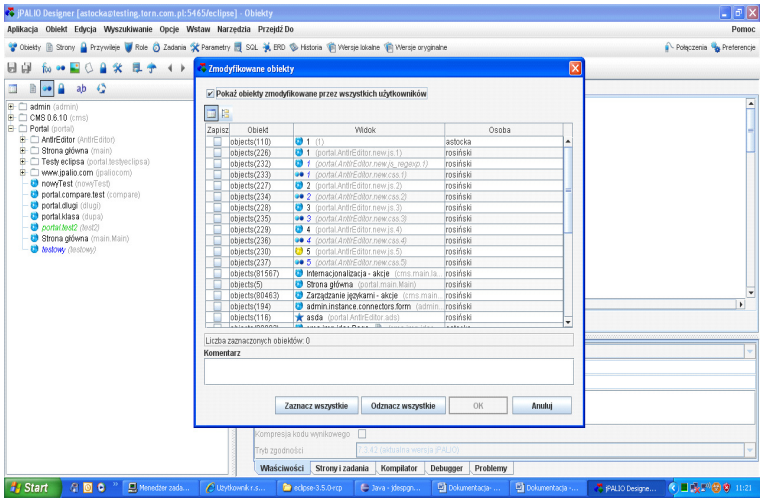
Wybrano jeden z obiektów spośród dostępnej listy obiektów - wybrany obiekt jest domyślnie zaznaczony jako obiekt, który ma zostać zarchiwizowany.



2.2. Lista zmodyfikowanych obiektów

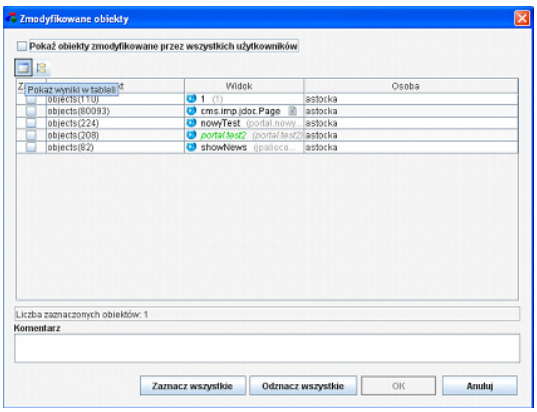
Aby stworzyć rewizję należy wybrać co najmniej jeden obiekt oraz dodać komentarz.

Domyślnie pokazywane są jedynie obiekty, zmodyfikowane przez użytkownika, który chce stworzyć rewizję. Istnieje również możliwość, pokazania wszystkich zmodyfikowanych obiektów, wystarczy jedynie wybrać opcję „Pokaż obiekty zmodyfikowane przez wszystkich użytkowników”, co zostało pokazane na rysunku poniżej.



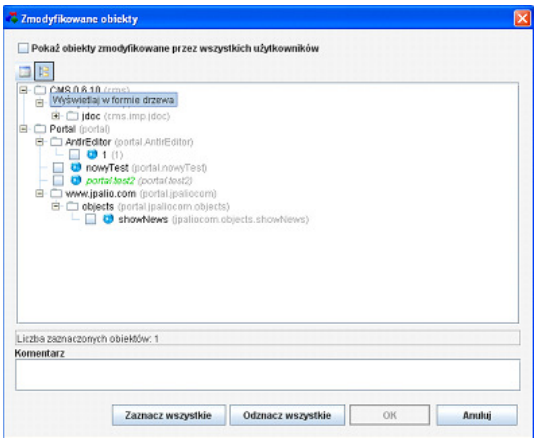
Lista zmodyfikowanych obiektów może zostać przedstawiona na dwa sposoby:

- w postaci tabeli

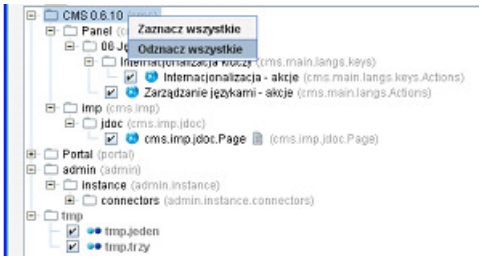


- w postaci drzewa

W przypadku drzewa widoczne są zarówno zmodyfikowane obiekty, jak i obiekty znajdujące się w hierarchii nad nimi, dzięki czemu łatwiej jest odnaleźć konkretny obiekt, jak również zarządzanie obiektami.



Kliknięcie prawym przyciskiem myszy na folder, powoduje pokazanie się menu, które umożliwia automatyczne zaznaczenie/odznaczenie wszystkich obiektów, które znajdują się w tym folderze.

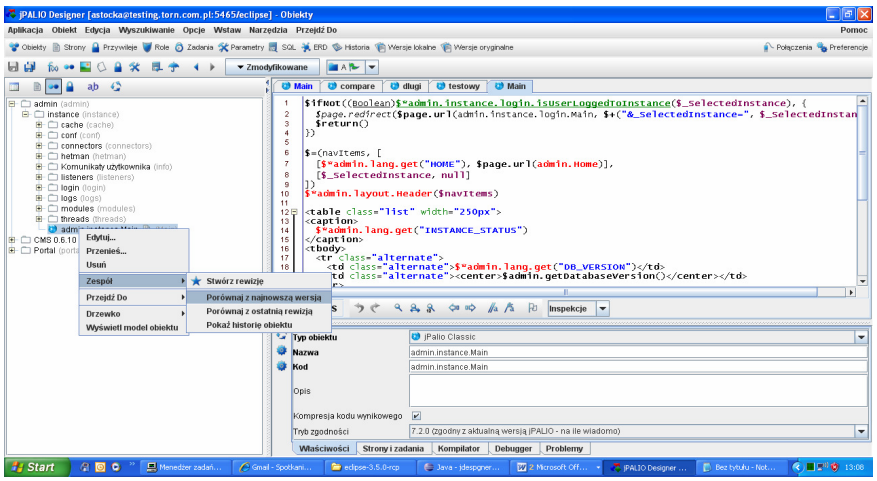


W celu odróżnienia obiektów usuniętych od istniejących, zostały one wyróżnione kolorem szarym. Przykład został zaprezentowany na rysunku powyżej.

2.3. Porównanie rewizji

W celu porównania różnych wersji obiektu należy kliknąć prawym przyciskiem myszy na zaznaczony obiekt, co spowoduje pojawienie się menu, następnie wybrać opcję „Zespół", po czym wybrać jedną z dostępnych opcji:

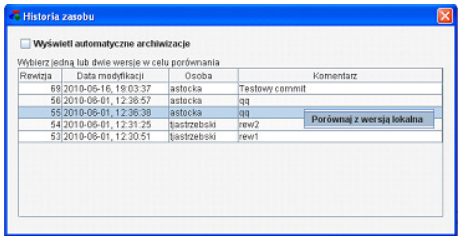
1. „Porównaj z najnowszą wersją" – powoduje porównanie lokalnej wersji obiektu, z ostatnią archiwalną wersją obiektu;
2. „Porównaj z ostatnią rewizją" - powoduje porównanie lokalnej wersji obiektu, z ostatnią archiwalną wersją obiektu, która była w rewizji;
3. „Pokaż historię obiektu" – powoduje wyświetlenie okna zawierającego historię danego obiektu.



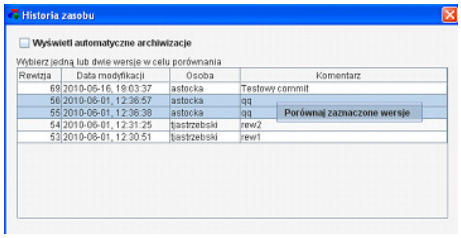
W oknie historii obiektu domyślnie wyświetlane są jedynie modyfikacje, które były w zawarte w rewizji. Zaznaczenie opcji „Wyświetl automatyczne archiwizacje" powoduje, że wyświetlone zostaną również automatyczne archiwizacje danego obiektu.



W Przypadku zaznaczenia jednej archiwalnej wersji oraz wciśnięcia prawego przycisku myszy, pojawia się menu z „Porównaj z wersją lokalną”, które powoduje porównanie zaznaczonej wersji obiektu, z wersją lokalną.

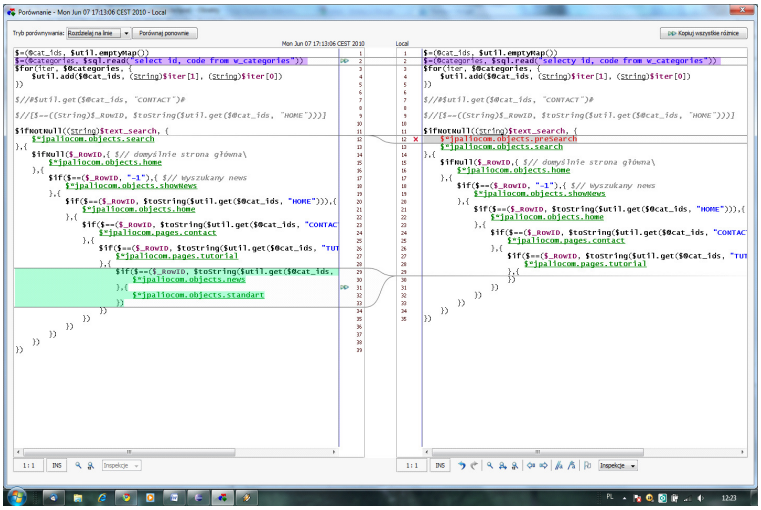


W Przypadku zaznaczenia dwóch archiwalnych wersji oraz wciśnięcia prawego przycisku myszy, pojawia się menu z „Porównaj zaznaczone wersje”, które powoduje porównanie zaznaczonych wersji archiwalnych.



3. Okno porównywania obiektów

Okno zbudowane jest z dwóch, sąsiadujących ze sobą edytorów: lewego – pokazującą archiwalną treść obiektu oraz prawego zawierającego treść aktualną. Nad edytorami widnieją podpisy pokazujące która wersja obiektu znajduje się w edytorze.

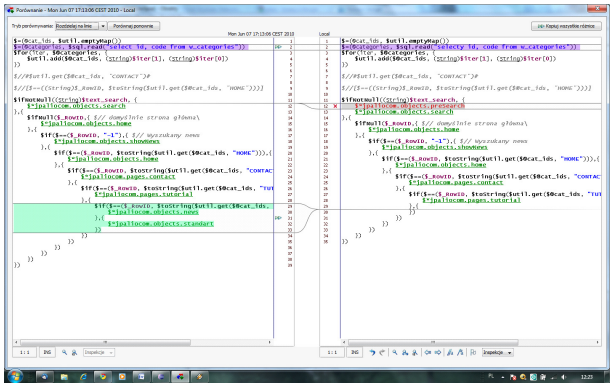


Prawy edytor z wersją lokalną jest zintegrowany z głównym edytorem designera, tzn. zmiany w nim dokonywane automatycznie dokonują się także w głównym edytorze programu, a zapis treści jest równoznaczny z zapisem z poziomu okna głównego.

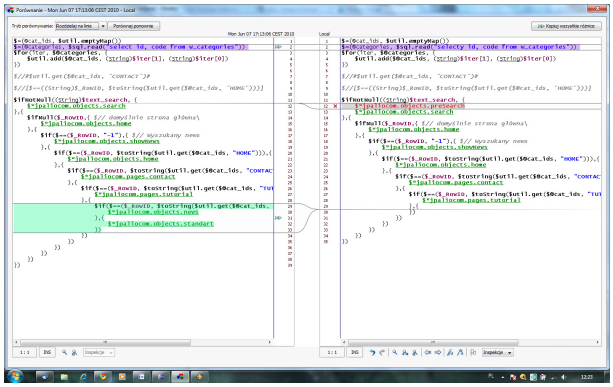
3.1. Markery podświetlające różnice

Podstawową funkcjonalnością okna porównywania jest pokazanie różnic w treściach dwóch różnych wersjach obiektu. Graficznie różnice reprezentowane są poprzez półprzezroczyste markery podświetlające odpowiadające sobie fragmenty kodu. W zależności od rodzaju zmiany dokonanej w kodzie wyróżnia się 3 podstawowe różnice oznaczane odrębnymi markerami:

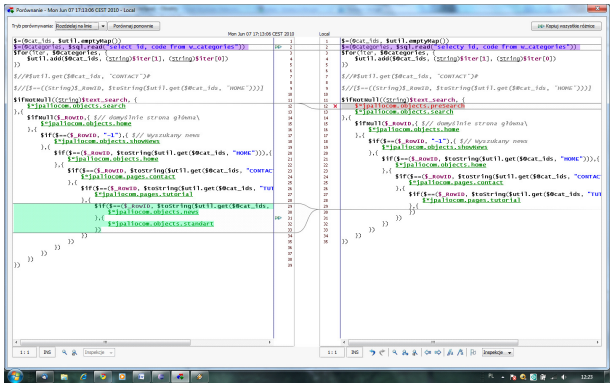
- Dodanie – W wersji lokalnej znajduje się fragment kodu, który nie występował w zarchiwizowanej treści. Możliwe jest usunięcie dodanego fragmentu poprzez wciśnięcie przycisku. Dodany kod jest zaznaczany kolorem szarym



- Usunięcie – W wersji lokalnej nie występuje już fragment, który występował w treści zarchiwizowanej. Fragment ten w lewym edytorze zaznaczany jest kolorem zielonym i możliwe jest skopiowanie go do wersji lokalnej poprzez wciśniecie przycisku.



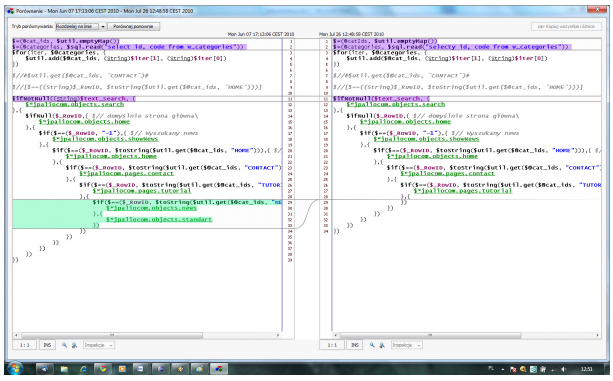
- Edycja – fragment występuje zarówno w wersji lokalnej jak i zarchiwizowanej, ale różni się treścią. Fragmenty w obydwu edytorach są zakreślone na niebiesko. Akcja dostępna na tym markerze poprzez wciśniecie przycisku to zastąpienie w wersji lokalnej różniącego się fragmentu odpowiednikiem z wersji historycznej.



W prawym górnym rogu okna porównywania obiektów widnieje przycisk „Kopiuj wszystkie różnice”. Służy on do usuwania wszystkich różnic poprzez przeniesienie zarchiwizowanej zawartości (lewego edytora) to lokalnej wersji obiektu. Po wciśnięciu przycisku cała treść lokalnej wersji obiektu zastępowana jest przez zarchiwizowaną umieszczoną w lewym edytorze. Funkcjonalność ta, może zatem służyć do przywracania archiwalnych wersji obiektu.

3.2. Porównywanie dwóch wersji archiwalnych

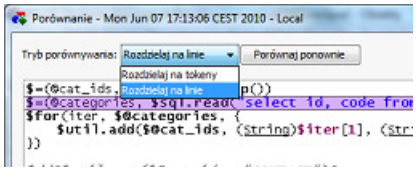
Okno to może służyć także nie tylko do porównywania ze sobą wersji lokalnej i archiwalnej ale także dwóch wersji archiwalnych. W tym wypadku istnieje jedynie możliwość podglądu treści obiektu w danych wersjach i różnic między nimi bez możliwości wprowadzania zmian w zarchiwizowanych treściach. Nie pojawiają się zatem przyciski pozwalające na przenoszenie treści z lewego edytora do prawego, prawy edytor jest w trybie tylko do odczytu a przycisk „Kopiuj wszystkie różnice” jest nieaktywny.



3.3. Mechanizm porównywania

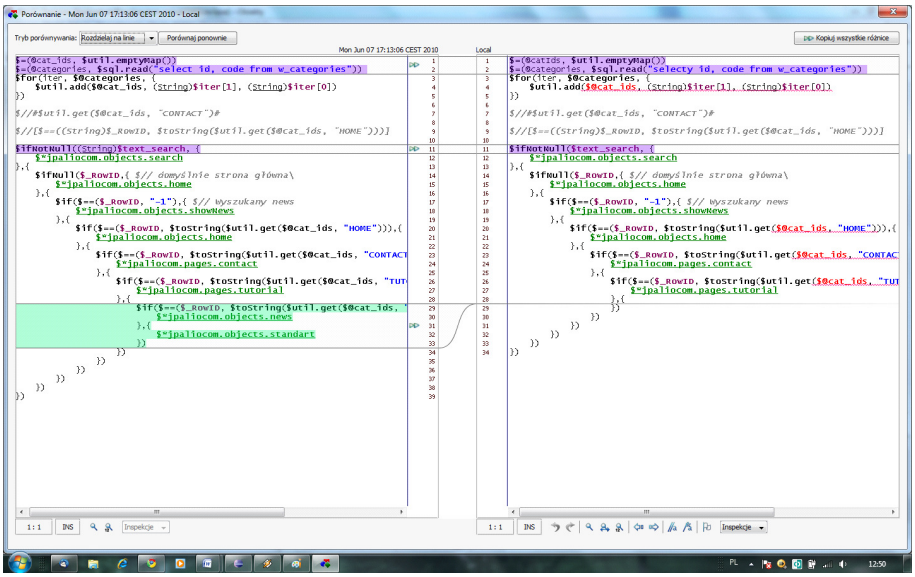
Porównywanie zawartości obiektów jest dokonywane jeszcze przed wyświetleniem na ekranie i nie jest na bieżąco aktualizowane, tzn. dokonane w prawym edytorze zmiany w wersji lokalnej nie są już aktualizowane i nowe różnice już nie są podświetlane. Aby zobaczyć nowe różnice należy wcisnąć przycisk „Porównaj ponownie” umiejscowiony w lewym górnym rogu okna.

Przy ponownym porównywaniu można określić tryb porównywania, tzn. w jaki sposób algorytm ma dzielić kod na elementy do porównywania. Pojedynczy element ma taką właściwość, że odpowiadające sobie w dwóch wersjach elementy porównywane są ze sobą jak zwykłe ciągi tekstowe, tzn. nawet różnica w pojedynczym znaku elementu powoduje, że elementy traktowane są jako różne. Jedynie zgodność co do każdego znaku powoduje, że elementy uznawane są za tożsame.

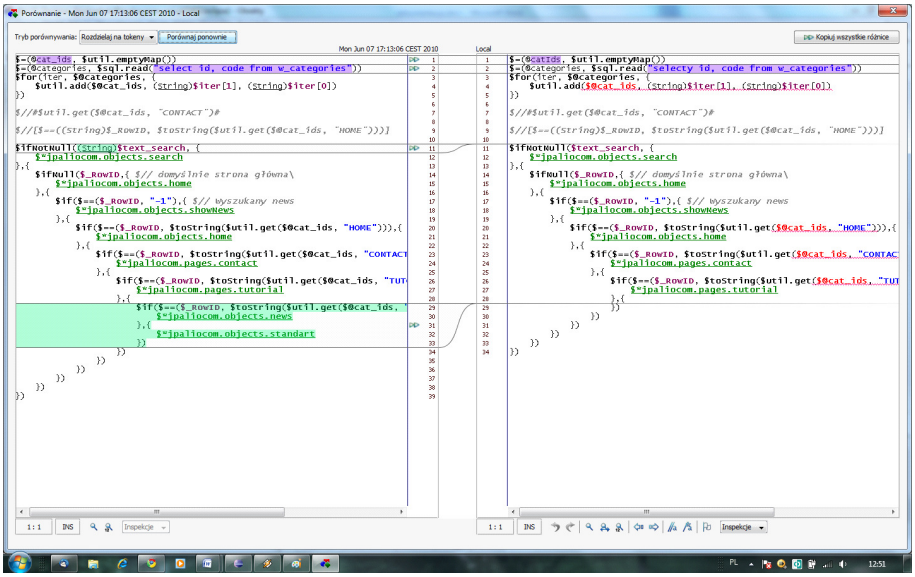


Aby wybrać tryb porównywania wystarczy z umiejscowionej obok przycisku „Porównaj ponownie” listy wybrać odpowiednią opcję. Dostępne są 2 tryby porównywania:

- Rozdzielaj na linie – domyślny tryb, pojedynczym elementem dla algorytmu jest cała linia tekstu. Nawet różnica na pojedynczym znaku w odpowiadających sobie liniach powoduje, że całe linie uznawane są za różne.



- Rozdzielaj na tokeny – pojedynczymi elementami są fragmenty kodu wydzielone na podstawie analizy leksykalnej kodu. Tokeny są zazwyczaj krótsze niż pojedyncza linia i odpowiadają elementom języka jPALIO, tzn. tokenem może być np. nazwa pojedynczej zmiennej, kod obiektu, ciąg tekstowy lub nazwa funkcji. Przy porównywaniu na tokeny należy uważać, aby porównywać kompilujące się kody. Jako, że do podziału na tokeny używa się analizy składni błąd składni może spowodować, że zostaną rozpoznane zupełnie różne tokeny a co za tym idzie mechanizm porównywania wykryje wiele różnic, które w rzeczywistości mogą nie występować (Szczególnie w przypadku niedomknięcia nawiasu bądź cudzysłowie).



4. Panel administracyjny

Panel administracyjny jest „bogatą” wersją znanej już wcześniej strony administracyjnej html.info. Ze względu na brak autoryzacji, zaleca się wyłączenie dostępu do starej strony administracyjnej html.info (można to zrobić usuwając odpowiednie wpisy z web.xml - pliku konfiguracyjnego Tomcat'a lub innego containera). Dodatkowo, panel posiada kilka nowych funkcjonalności.

4.1. Dostęp

Panel działa podobnie jak zwykła aplikacje webowa. Aby uruchomić aplikację należy tylko w przeglądarce wybrać adres zgodny z wzorcem:

http://adres_serwera/palio/html.run ,gdzie:

- adres serwera to nazwa hosta (wraz z opcjonalnym portem podanym po dwukropku),
- palio (nazwa kontekstu Tomcata lub innego containera).

4.2. Autoryzacja

Dostęp do panelu wymaga uwierzytelniania i jedynie autoryzowani użytkownicy mogą z niego korzystać. Wyróżnia się dwa poziomy autoryzacji:

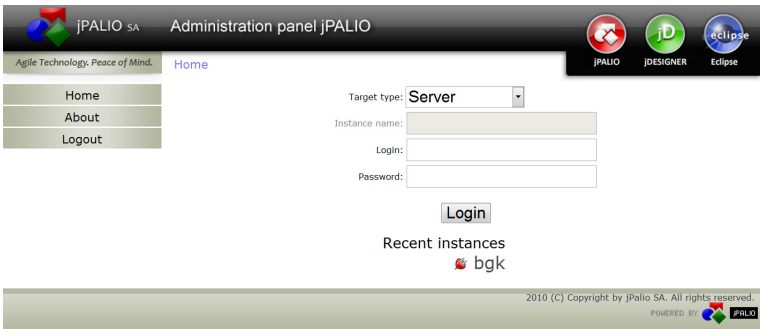
- Logowanie do serwera – Po zalogowaniu do serwera mamy pełny dostęp do zarządzania serwerem oraz dostęp do zarządzania wszystkimi instancjami. Aby zalogować się do serwera na stronie logowania należy wybrać z listy „Target type” opcję Server oraz wpisać w odpowiednio login i hasło administratora serwera (więcej informacji znajdziesz w dokumentacji pliku konfiguracyjnego serwera jPALIO – config.xml).



- Logowanie do instancji – Panel umożliwia także zalogowanie się bezpośrednio do instancji co umożliwia zarządzanie jedynie tą instancją do której jesteśmy zalogowani a opcje zarządzania serwerem oraz pozostałymi instancjami są ukryte. Do logowania można używać loginu i hasła używanego w aplikacji jDESIGNER (wybrany użytkownik musi posiadać rolę „designer"). Aby zalogować się do konkretnej instancji należy w oknie logowania z listy „Target type" wybrać opcję Instance, w polu „Instance name" wpisać nazwę instancji (ze względów bezpieczeństwa lista instancji nie jest wyświetlana użytkownikom nie zarejestrowanym do serwera) a w polu login i hasło użytkownika aplikacji jDESIGNER.

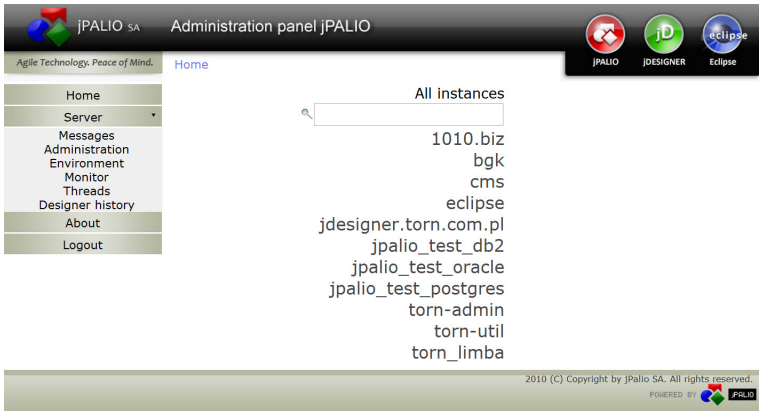
Możliwe jest także aby strona logowania domyślnie ustawiona była na logowanie do konkretnej instancji. Wystarczy w url ustawić parametr _SelectedInstance=nazwa_instancji, np. adres http://adres_serwera/palio/html.run?_SelectedInstance=foo automatycznie otworzy stronę logowania do instancji foo.

Użytkownik logujący się do bezpośrednio do wybranej instancji (nie logujący się na serwer), może w ramach jednej sesji zalogować się do dowolnej liczby instancji. Logowanie się do innej instancji bądź przełączanie się między otwartymi instancji odbywa się poprzez zakładkę Home.



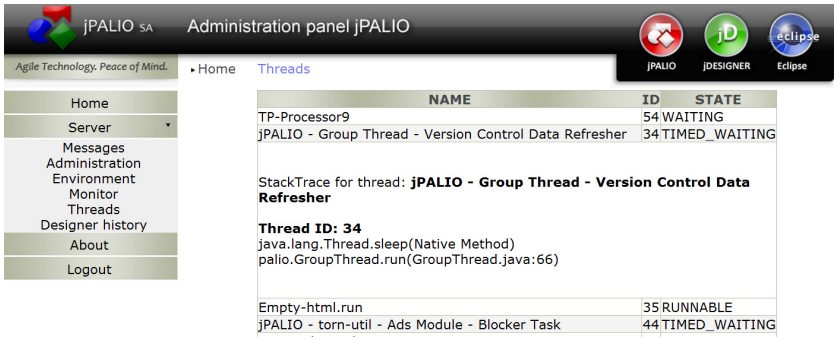
4.3. Zarządzanie serwerem

Po zalogowaniu się do serwera pojawia się lista wszystkich instancji zainstalowanych na serwerze. Po kliknięciu w instancję przechodzi się do strony zarządzania daną instancją (Szczegółowo zarządzanie instancją zostanie opisane później).

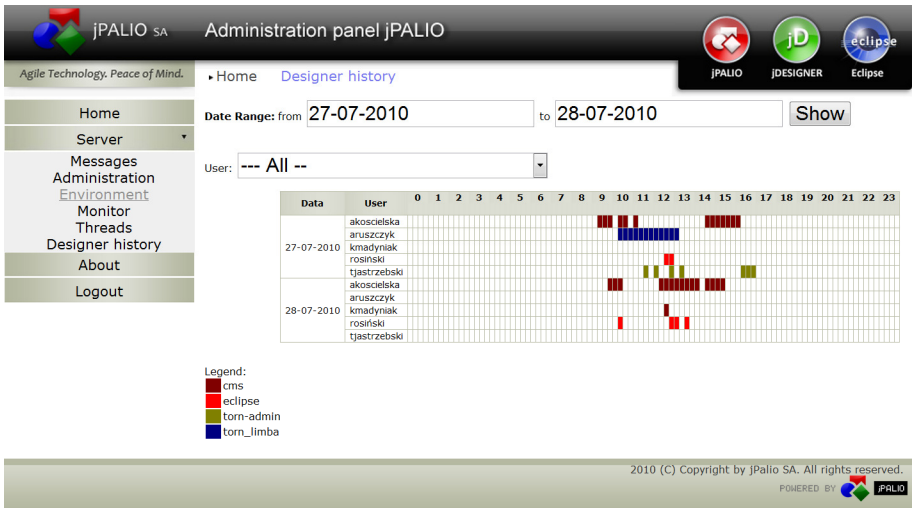


Oprócz listy instancji w lewym bocznym menu dostępne są różne zakładki pozwalające wykonywać konkretne czynności na serwerze.

- Messages – lista wiadomości serwera.
- Administration – Akcje administracyjne, aktualnie znajduje się na niej przycisk „Rescan instances directory" wymuszający na serwerze, ponowne prze skanowanie katalogu z konfiguracjami instancji i wystartowanie nowych instancji (bez restartu serwera).
- Environment – Strona informacyjna zawierająca podstawowe dane środowiska pracy serwera.
- Monitor – zawiera listę klas Groovy dostępnych na serwerze oraz liczbę ich wersji (zapisanie klasy Groovy w aplikacji jDESIGNER tworzy nową wersję tej klasy na serwerze, niektóre dane jednak mogą nadal odwoływać się do „starszej" wersji klasy).
- Threads – lista wszystkich wątków serwera. Po kliknięciu w nazwę wyświetla się stack trace dla wątku.



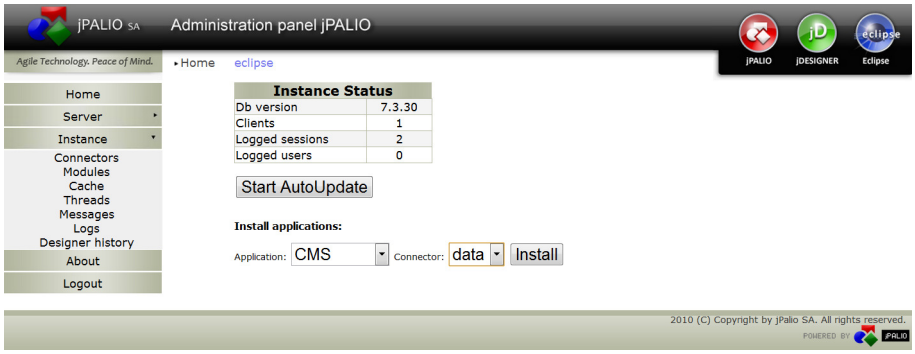
- Designer history – moduł śledzenia pracy developerów na wybranych instancjach. Aktywność użytkowników na konkretnych instancjach jest wyświetlana w postaci siatki z piętnastominutową dokładnością. Możliwe jest także przefiltrowanie wykresy za pomocą umiejscowionej nad wykresem listy User i wyświetlenie aktywności jedynie wybranego użytkownika. Dodatkowo po najechaniu kursorem na za kolorowaną komórkę wykresu pojawią się szczegółowe informacje o danej aktywności.



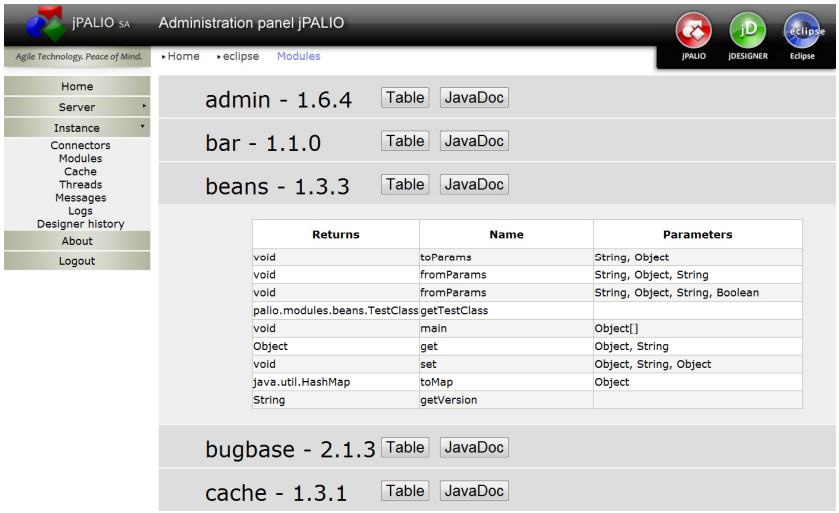
4.4. Zarządzanie instancją

Głównym celem funkcjonalności oferowanej przez panel administracyjnym w obszarze zarządzania instancją jest dopełnianie funkcjonalności oferowanej przez aplikację jDESIGNER.

- Strona główna zarządzania instancją – oprócz tabeli informacyjnej dostępny na stronie jest przycisk „Start AutoUpdate”, który pozwala wymusić ponowne wykonanie automatycznej aktualizacji bazy danych instancji (skanowanie istniejącej struktury i uzupełnienie „braków”). Na stronie znajduje się także formularz „Install application”, który umożliwia stworzenie struktury bazy danych pod wybraną aplikację na wybranym konektorze. Dostępne są 3 aplikacje: CMS – Content Management System, Newsletter – aplikacja umożliwiająca użytkownikom mailową subskrypcję informacji oraz Ads – system reklamowy. Aplikacja CMS wymaga zainstalowanych wszystkich struktur CMS, Newsletter i Ads. Dostęp do strony można uzyskać zawsze po kliknięciu na główną opcję menu Instance.



- Connectors – wyświetla listę konektorów oraz umożliwia nimi zarządzanie.
- Modules – tabela modułów dostępnych w ramach instancji. Możliwy jest podgląd funkcji dostępnych w ramach każdego modułu oraz podgląd dokumentacji modułu w postaci JavaDoc.



- Cache – moduł zarządzania cache'ami instancji. Pozwala opróżniać cache pojedynczo oraz wszystkie jednocześnie.
- Threads – tabela wątków analogiczna jak przy zarządzaniu serwerem, różnicą jest to, że tutaj mamy dostępne jedynie wątki danej instancji.
- Messages – lista wiadomości instancji
- Logs – interaktywne wyświetlanie logów instancji przy użyciu technologii AJAX (webowy odpowiednik polecenia tail znanego z Linux / Unix).
- Designer history – moduł śledzenia historii podobny do modułu „Designer history" przy zarządzaniu serwerem, jednak tutaj śledzona jest jedynie aktywność na konkretnej instancji którą zarządzamy.

5. Zarządzanie aktywnymi łatkami

W wersji jPALIO 7.4 zarządzanie łatkami zostało przeniesione do modułu „Obiekty". W panelu znajdującym się w górnej części okna, został dodany przycisk, wyświetlający łatkę, która jest w chwili obecnej aktywna.

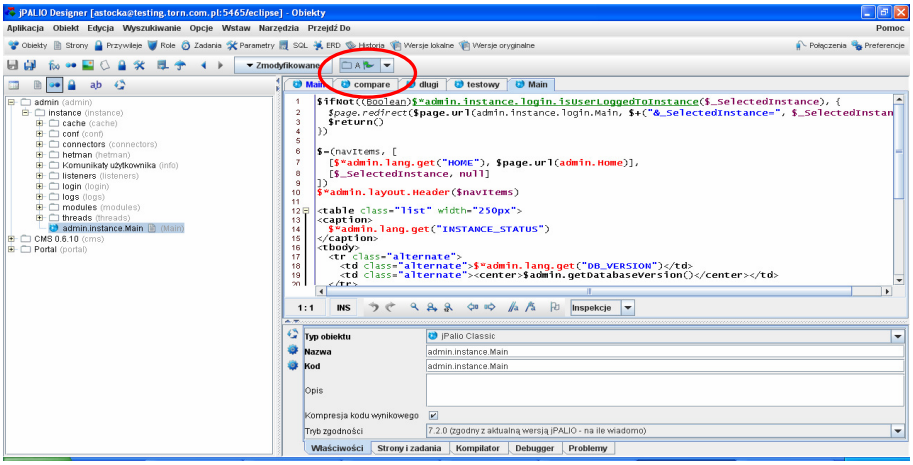
5.1. Zarządzanie aktywnymi łatkami w module „Obiekty"

W zależności od wybranej łatki możliwe jest otrzymanie następujących rezultatów:

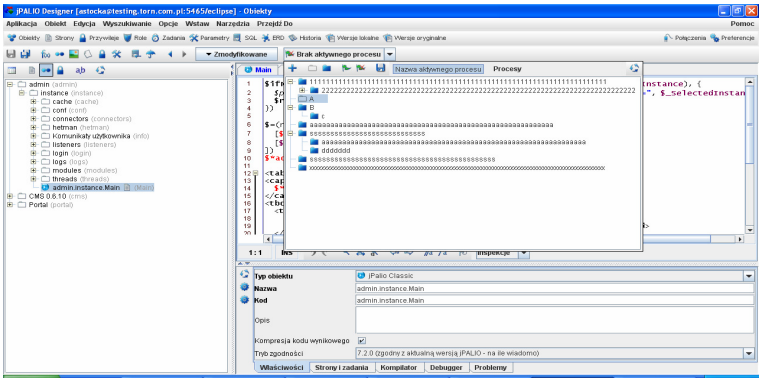
- Nazwa łatki - w przypadku gdy istnieje aktywna łatka.

- „Brak aktywnego procesu” - w przypadku gdy nie istnieje aktywna łatka.
- Nazwa łatki na czerwonym tle, w przypadku próby ustawienia zamkniętej łatki jako aktywną.

Opisany przycisk został zaprezentowany na obrazku poniżej:

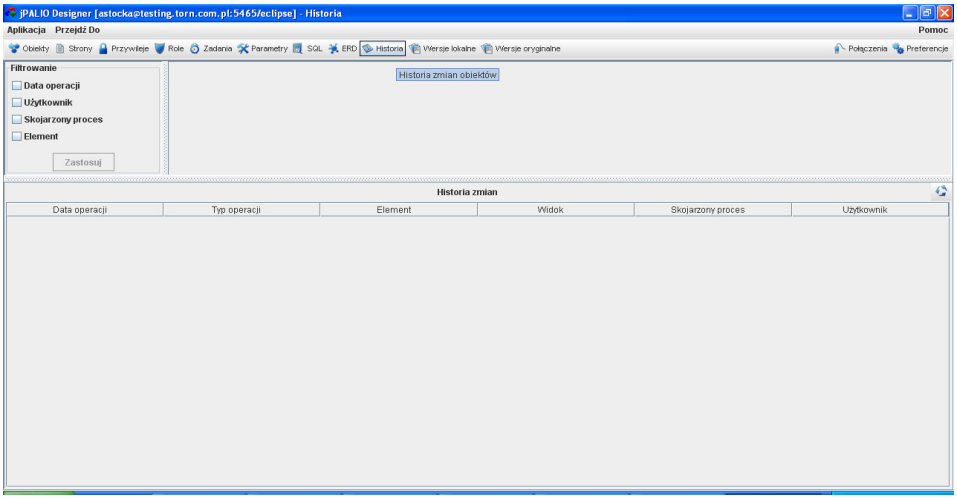


Przyciśnięcie przycisku, powoduje wyświetlenie okna zawierającego listę dostępnych łatek jak również zmianę aktywnej łatki.



5.2. Zmiana nazwy modułu „Projekt” na „Historia”

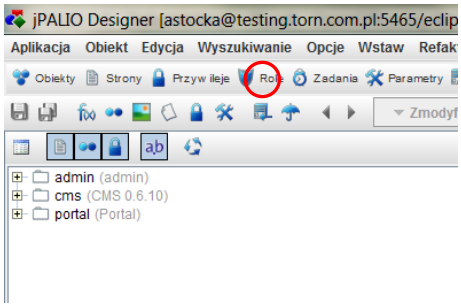
Po przeniesieniu zarządzania aktywnymi łatkami do modułu „Obiekty”, moduł „Projekt” zawiera już wyłącznie panel przeglądania historii. W związku z tym moduł „Projekt” zmienia nazwę na „Historia”.



6. Wgrywanie łatek

W nowej wersji serwera rozszerzono sposób wgrywania łatek. Okno parametrów wgrywania łatek zostało podzielone na trzy zakładki, każda działająca niezależnie. Zakładka „Kompatybilne” zawiera konfigurację znaną z poprzednich wersji. Pozostałe dwie nowo dodane zakładki opisują pełną zaawansowaną oraz uproszczoną konfigurację. Główną cechą wprowadzonych zmian jest możliwość określenia podejmowanej akcji dla każdego typu elementów niezależnie (np. można wgrać role, przywileje, połączenia między rolami i przywilejami a bez wgrywania stron). Dodatkowo dla obiektów oraz multimediów dodano do wyboru nowy rodzaj akcji „Wgraj nadpisując tylko starsze” (akcja ta powoduje zastąpienie elementu przez element w łatce tylko wtedy gdy data modyfikacji elementu z łatki jest nowsza niż data modyfikacji na serwerze docelowym). Dodatkowo istnieje możliwość automatycznej aktywacji/dezaktywacji wgranych z łatki zadań.

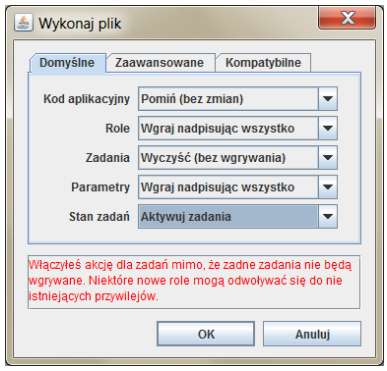
Jednocześnie w związku z przeniesieniem zarządzania aktywnymi łatkami do modułu „Obiekty”, akcja wgrywania łatek została również przeniesiona do tego modułu.



6.1. Zadkładka „Domyślne”

Zakładka „Domyślne” jest domyślną, uproszczoną formą nowej konfiguracji. Zgodnie z nową konfiguracją (podobnie jak w zakładce „Zaawansowane”), użytkownik musi określić oczekiwane akcje do podjęcia dla każdego z typów wgrywanych elementów. W przypadku zakładki „Domyślne” opcje są zgrupowane w formie wygodnej dla standardowych użytkowników:

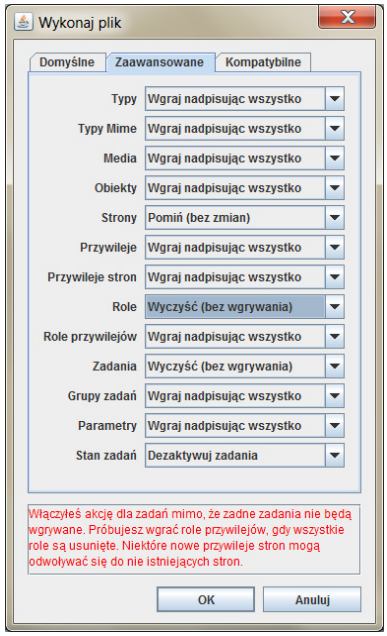
- Opcja „Kod aplikacyjny” jest opcją zbiorczą dla elementów: katalogi, typy MIME, obiekty, multimedia, strony, przywileje oraz przypisania przywilejów do stron,
- opcja „Role” odpowiada za role i przypisania przywilejów do ról,
- opcja „Zadania” odpowiada za zadania i grupy zadań,
- opcja „Parametry” dotyczy tylko parametrów globalnych (dokładnie jak w zakładce „Zaawansowane”),
- opcja „Stan zadań” określa dodatkową akcję jaka może być podjęta dla wgranych zadań (aktywacja/dezaktywacja/pozostawienie bez zmian – stan zgodny z łatką).



Wybranie akcji „Wgraj nadpisując tylko starsze” w opcji „Kod aplikacyjny” dotyczy wyłącznie obiektów oraz multimediów. Dla pozostałych elementów akcja ta jest równoznaczna z akcją „Wgraj nadpisując wszystko”.

6.2. Zakładka „Zaawansowane”

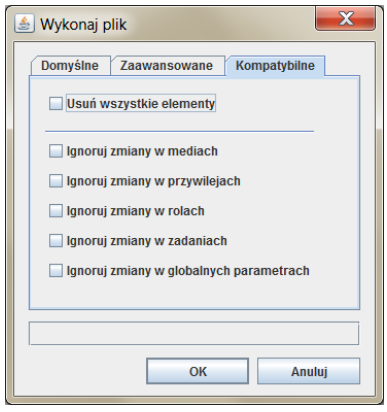
Zakładka „Zaawansowane” pozwala na wybranie niezależnych akcji dla każdego typu elementów (jak na obrazku poniżej). Użytkownik jest odpowiedzialny za wybranie takiej konfiguracji, która jest „sensowna”. Ponieważ wiele możliwych jest potencjalnie błędna, warto zwracać uwagę na pisane czerwoną czcionką wskazówki wyświetlane w dolnej części okna.



Poza nie łączeniem elementów w grupy elementów, zakładka jest w pełni zgodna z zakładką „Domyślne”.

6.3. Zakładka „Kompatybilne”

Zakładka została przeniesiona w niezmienionej formie z okna wgrywania łatek z poprzednich wersji jDESIGNER'a. Celem istnienia łatki jest zapewnienie ciągłości bez wymuszania na użytkownikach nauki nowej funkcjonalności. Zaleca się jednak używanie zakładek „Domyślne” lub „Zaawansowane” ze względu na bardziej rozbudowaną konfigurację.



7. Automatyczna aktualizacja

jPALIO 7.4 wprowadza znaczną rozbudowę mechanizmu AutoUpdate. Od tego momentu każdy proces aktualizacji struktur bazy danych jest poprzedzany skanowaniem aktualnej struktury. Tak więc proces aktualizacji może dokonać nie tylko przewidzianej modyfikacji struktur ale również uzupełnić ewentualne braki. Ze względu na to, użytkownicy ingerujący w systemowe tabele powinni odpowiednio skonfigurować proces AutoUpdate, które tabele/sekwencje powinny zostać pominięte.

7.1. Konfiguracja instancji

Aby oznaczyć które tabele lub sekwencje powinny być pominięte należy dodać następujące wpisy w konfiguracji instancji. Dla wskazanych tabel – poza samymi tabelami – będą również pomijane ograniczenia (klucze obce) wskazujące na te tabele.

```
<autoupdate>
    <table name="P_OBJECTS" action="skip"/>
    <sequence name="P_OBJECTS_S" action="skip"/>
</autoupdate>
```

W większości przypadków w konfiguracji instancji konfiguracja AutoUpdate może zostać pominięta.

7.2. Tworzenie/aktualizacja struktur dla zewnętrznych aplikacji

Proces AutoUpdate został rozszerzony na inne niesystemowe struktury. Może być użyty do utworzenia struktur dla aplikacji CMS, Newsletter oraz Ads – wywołanie AutoUpdate dla tych aplikacji odbywa się za pomocą panelu administracyjnego.

7.3. Zasady działania AutoUpdate

Algorytm automatycznej aktualizacji oparty jest na kilku zasadach:

- dodane z zewnątrz nieznane tabele lub kolumny nie będą usuwane,
- tylko wybrane tabele lub kolumny (które w ramach modyfikacji struktur specjalnie oznaczono do usunięcia) będą usuwane,
- brakujące tabele, kolumny, ograniczenia będą odtwarzane,
- rozmiar kolumn może być rozszerzany, nigdy natomiast nie będą skracane – nawet gdy przewidziany rozmiar jest mniejszy niż aktualny,
- ograniczenia na kolumnach kolidujące z oczekiwanymi ograniczeniami mogą zostać usunięte i zastąpione docelowymi ograniczeniami.

8. Konfiguracja serwera

8.1. Konto administratora serwera

Konto administratora serwera jest używane do zarządzania serwerem. W panelu administracyjnym można zalogować się do bezpośrednio do wybranej instancji lub na serwer (w takim przypadku mamy dostęp do wszystkich instancji). Konto administratora serwera definiujemy w pliku config.xml. Hasło do konta można podać w pliku konfiguracyjnym w postaci jawnej lub zakodowanej. Jeżeli zdecydujesz się na postać zakodowaną, w celu wygenerowania hasła należy użyć wybranej funkcji skrótu a następnie użyć kodowania Base64. W przypadku użycia funkcji skrótu, należy w konfiguracji podać nazwę użytego algorytmu (SHA1, MD5, itd.). W przypadku pominięcia algorytmu, jako hasło można podać bezpośrednio wybrane hasło w jawnej postaci (bez Base64)

```
<admin login="admin" password="password" algorithm="sha1"/>
```

Zakodowane hasło można łatwo uzyskać stosując np. komendę:

```
echo -n "hasło" | openssl dgst -sha1 -binary | openssl base64
```

8.2. Wybór uruchamianej wersji jDESIGNER'a

Kiedy klikamy na link jDESIGNER obok nazwy instancji w panelu administracyjnym (lub html.info), uruchamia się odpowiednia wersja aplikacji jDESIGNER. Do tej pory mapowanie między wersją serwera a wersją klienta było jawne i tworzone arbitralnie. Jednakże od wersji serwera 7.4, numer wersji klienta jDESIGNER jest dobierany automatycznie. Ze względu na to, dajemy Ci możliwość wpływania na to, jaka wersja aplikacji zostanie uruchomiona dla Twojego serwera. W konfiguracji serwera możesz dodać następujące wpisy:

```
<designer version="latest"/>
```

jeżeli chcesz aby była uruchamiana zawsze najnowsza wersja aplikacji lub

```
<designer version="stable"/>
```

jeżeli chcesz aby była uruchamiana stabilna wersja aplikacji. W przypadku pominięcia argument „version” domyślnym wyborem jest użycie wersji stabilnej.

jPALIO SA
ul. Joteyki 20
02-317 Warszawa

Tel: +48 22 424 87 88
Fax: +48 22 313 23 37

Email: kontakt@jpalio.com

www.jpalio.com