

12/3/24, 11:01 утра



# Коммерция 123

Сгенерировано: 2024-12-03 11:01:21 GMT+0000

SAP Коммерция | 2205

Публичный

Оригинальное содержание:[https://help.sap.com/docs/SAP\\_COMMERCE/3fb5dcfe37f40edbac7098ed40442c0?locale=en-US&state=PRODUCTION&version=2205](https://help.sap.com/docs/SAP_COMMERCE/3fb5dcfe37f40edbac7098ed40442c0?locale=en-US&state=PRODUCTION&version=2205)

Предупреждение

Этот документ был создан на основе SAP Help Portal и является неполной версией официальной документации по продукту SAP. Информация, включенная в пользовательскую документацию, может не отражать расположение тем на SAP Help Portal и может не иметь важных аспектов и/или корреляций с другими темами. По этой причине он не предназначен для продуктивного использования.

Для получения более подробной информации посетите сайт<https://help.sap.com/docs/ отказ от ответственности>.

# Изучите тур SAP Commerce

В туре «Изучение SAP Commerce» вы используете основные функции SAP Commerce для создания веб-сайта для вымышленной компании под названием Little Concert Company.

## Примечание

Если вы проходите несколько экскурсий, удалите все разархивированные папки SAP Commerce и перезагрузите компьютер между экскурсиями, чтобы убедиться, что нет запущенных потоков SAP Commerce. Затем выполните шаги в [Прежде чем начать](#) и начните свой следующий тур.

Целью является удовлетворение реалистичных, но вымышленных деловых потребностей Little Concert Company. Little Concert Company продает билеты на электрический ассортимент небольших независимых концертных туров. Каждый тур состоит из ряда концертов, проводимых в небольшом наборе мест по всей Европе. В настоящее время билеты продаются по телефону, но теперь компания хочет добавить эту возможность на свой веб-сайт, заменив старый, заказной веб-сайт и ручной процесс бронирования в бэк-офисе на современную коммерческую систему на основе SAP Commerce.

Пройдите разделы по порядку, прочитайте немного о представленных концепциях и примените их на практике.

## [Контроль версий](#)

Системы контроля версий являются неотъемлемой частью любого нетривиального проекта разработки программного обеспечения. Основная цель системы контроля версий — централизованное отслеживание и хранение изменений, вносимых в программное обеспечение по мере его разработки.

## [Рецепты установщика](#)

SAP Commerce поставляется с набором предварительно настроенных скриптов установки, называемых рецептами установщика. Рецепты установщика упрощают установку и настройку SAP Commerce для целей разработки или демонстрации.

## [Расширения](#)

SAP Commerce имеет модульную архитектуру, в которой новая бизнес-логика разрабатывается в отдельных функционально-специфичных модулях, называемых расширениями.

## [Файл localextensions.xml](#)

The localextensions.xml file содержит список расширений, которые ваша конкретная конфигурация SAP Commerce включает во время компиляции и выполнения.

## [Модели данных](#)

Модель данных, лежащая в основе SAP Commerce, определяется в XML-файлах. Новые типы данных для расширений, называемые типами элементов, определяются в <имя-расширения>-items.xml\les.

## [Проектирование баз данных](#)

SAP Commerce хранит свои данные в реляционной базе данных. Поддерживается большой спектр сторонних баз данных, что позволяет вам выбрать базу данных, наиболее подходящую для вашего решения.

## [ИмпЭкс](#)

SAP Commerce поставляется с мощными функциями импорта и экспорта текстовых данных под названием ImpEx.

## [Основные и проектные данные по соглашению](#)

Вы можете подготовить файлы ImpEx, содержащие основные и проектные данные, которые система импортирует при инициализации. Эта функциональность следует принципу Convention over Configuration (CoC).

## [Основные и проектные данные по коду](#)

Вы можете подключиться к процессу инициализации и обновления системы, чтобы загрузить данные проекта во время инициализации платформы.

## [Основные и проектные данные по конвенции и кодексу](#)

SAP Commerce ищет и загружает данные из файлов ImpEx, которые следуют определенному соглашению об именовании. Такое поведение поддерживает парадигму проектирования программного обеспечения «соглашение поверх конфигурации».

## [Уровень обслуживания](#)

При реализации новой бизнес-логики вы разделяете бизнес-код на классы Java, называемые сервисами. Каждый сервис реализует определенное, четко определенное требование.

## [Интеграционные тесты](#)

Интеграционные тесты необходимы для демонстрации того, что ваша новая функциональность работает так, как и ожидалось. Они уведомляют вас, когда вы нарушаете существующее поведение, и, следовательно, могут помочь сократить количество ошибок.

## [Тесты модулей](#)

Вы можете моделировать зависимости для выполнения модульных тестов, которые выполняются независимо от платформы SAP Commerce.

## [Фасадный слой](#)

Фасад — это уровень абстракции, который предоставляет упрощенный интерфейс для базовой реализации.

## [Передняя часть](#)

После того, как у вас есть модель и бизнес-логика, вы можете разработать подходящее веб-приложение front-end. При создании front-end используйте фреймворк Spring MVC для разделения частей модели, представления и контроллера.

## [Динамические Атрибуты](#)

Динамические атрибуты позволяют добавлять атрибуты в модель и создавать для них пользовательскую логику, не касаясь самого класса модели. Они предоставляют способ генерировать новые данные и получать к ним доступ без вызова отдельной службы для этого. Динамические атрибуты — это временные данные, которые не сохраняются в базе данных.

## [Интеграция динамических атрибутов](#)

Хорошей практикой является всегда проводить интеграционное тестирование при внедрении новых функций, чтобы тестировать новые классы в контексте.

## [Обновление веб-страницы](#)

Обновите соответствующие части вашего расширения, чтобы использовать новый динамический атрибут.

## [События и слушатели](#)

Система событий — это структура, предоставляемая ServiceLayer, который позволяет отправлять и получать события в SAP Commerce.

## [Перехватчики и пользовательские события](#)

Перехватчики перехватывают переходы жизненного цикла объектов модели и, в зависимости от условий этого перехода, могут публиковать событие, когда они это делают.

## [События, учитывающие кластер](#)

SAP Commerce поддерживает события, поддерживающие кластер. Благодаря событиям, поддерживающим кластер, SAP Commerce может обрабатывать события в отдельных потоках или на определенных узлах кластера.

## [Задания Cron](#)

SAP Commerce предоставляет средства для настройки регулярных задач. С помощью этих задач, или заданий cron, вы можете многократно выполнять сложную бизнес-логику в определенное время и с определенными интервалами.

## [Триггеры](#)

Успешно внедрив бизнес-логику в класс задания, вы можете запустить его выполнение с помощью задания cron.

## [Круто](#)

Для написания заданий для выполнения можно использовать язык сценариев Groovy.

## [Панель управления администрированием бэк-офиса](#)

Backoffice Administration Cockpit — это удобный пользовательский интерфейс на основе браузера для просмотра, создания и управления данными в SAP Commerce.

[Локализация](#)

Локализация направлена на адаптацию SAP Commerce к нескольким языкам.

[Локализация в административной панели бэк-офиса](#)

Вы можете определять локализованные значения для атрибутов типа элемента непосредственно в панели администрирования Backoffice.

[Проверка](#)

Фреймворк проверки данных SAP Commerce обеспечивает чистые, правильные и полезные данные. Фреймворк проверки основан на спецификации проверки Java, JSR 303. Он предлагает простой и расширяемый способ проверки данных перед их передачей на уровень сохранения.

[Ограничения проверки в бэк-офисе](#)

Вы можете создавать и определять ограничения проверки в панели администрирования SAP Commerce Backoffice.

[Ограничения проверки в ImpEx](#)

Вы можете определять ограничения проверки в файлах ImpEx, что упрощает повторную загрузку ограничений после инициализации базы данных.

[Пользовательские ограничения проверки](#)

Хотя фреймворк проверки предоставляет все ограничения из спецификации JSR 303, иногда вам могут потребоваться другие типы ограничений, специфичные для вашего проекта.

[Интеграционный тест для пользовательского ограничения](#)

Ознакомьтесь с тем, как использовать службу проверки SAP Commerce в коде.

[Медиа-файлы](#)

SAP Commerce поддерживает файлы носителей. Файл носителя может быть чем угодно, что может быть сохранено в файловой системе.

[Файлы свойств](#)

SAP Commerce опирается на два основных файла конфигурации: project.properties и local.properties. Свойства проекта являются значениями по умолчанию SAP Commerce, в то время как локальные свойства — это то место, где вы можете определить собственную конфигурацию для своего расширения.

## Контроль версий

Системы контроля версий являются неотъемлемой частью любого нетривиального проекта разработки программного обеспечения. Основная цель системы контроля версий — централизованное отслеживание и хранение изменений, вносимых в программное обеспечение по мере его разработки.

Как разработчик, вы, как правило, совершили изменения несколько раз в день, после завершения небольшого приращения работы. Эти коммиты должны быть сделаны только тогда, когда ваше программное обеспечение находится в зеленом состоянии. Другие разработчики в вашей команде могут затем тянуть изменения из системы контроля версий, что позволяет команде эффективно обмениваться любыми изменениями. Система контроля версий также позволяет вам откатывать код к более раннему коммиту, если вам это нужно.

В SAP Commerce 123 вы будете использовать систему контроля версий Git для фиксации кода по мере продвижения. Это позволит вам вернуться к ранее выполненному шагу, если вы этого захотите.

Родительская тема:[Изучите тип SAP Commerce](#)

Следующий:[Рецепты установщика](#)

Сопутствующая информация

[Git](#)

## Настройте Git-репозиторий

Создайте репозиторий Git, в который вы будете сохранять свой код по мере продвижения по SAP Commerce 123. Зафиксировав изменения в Git, вы сможете извлечь код ранее выполненных шагов, если это потребуется.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.java.

```
public void gitRepoOk() {
    String output = CommandLineHelper.runCmd("git --git-dir ..\hybris/.git log"); assertTrue("Репозиторий Git настроен неправильно",
        output.contains("Настройте репозиторий Git"));
}
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#gitRepoOk test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#gitRepoOk test
```

### Процедура

. Настройте репозиторий Git в папке <HYBRIS\_HOME\_DIR>/гибрис

```
cd $HYBRIS_HOME_DIR/hybris; git init
```

```
cd %HYBRIS_HOME_DIR%\hybris и git init
```

. Чтобы ограничить файлы, сохраняемые в вашем репозитории Git, добавьте файл с именем .gitignore в <HYBRIS\_HOME\_DIR>/гибрис спапка с содержимым:

```
/бревно  
/тэмп  
/bin/
```

/bin/пользовательский  
/роли

Интерактивный ярлык SAP Commerce 123: если вы не можете создать файл самостоятельно или хотите пропустить этот шаг, переместите <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/.gitignore в <HYBRIS\_HOME\_DIR>/гибрис

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/.gitignore $HYBRIS_HOME_DIR/hybris
```

```
xcopy /h /y %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\.\gitignore %HYBRIS_HOME_DIR%\hybris
```

.Сделайте начальный коммит в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add . ; git commit -m "Настройка репозитория Git"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Настроить репозиторий Git"
```

.Бегатьмерзавец бревно убедитесь, что вы видите в списке свой первый коммит git.

мерзавец бревно

мерзавец бревно

.Запустите gitRepoOk повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#gitRepoOk тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#gitRepoOk test
```

## Рецепты установщика

SAP Commerce поставляется с набором предварительно настроенных скриптов установки, называемых рецептами установщика. Рецепты установщика упрощают установку и настройку SAP Commerce для целей разработки или демонстрации.

Рецепты установщика создают каталоги, перемещают файлы, обновляют свойства и настройки конфигурации и инициализируют систему для определенной конфигурации SAP Commerce. С помощью одного вызова рецепта установщика вы можете настроить, инициализировать и запустить SAP Commerce в ряде вариаций, включая ускорители, интеграции SAP или минимальную конфигурацию только платформы SAP Commerce.

-Примечание

Не используйте установщик для запуска одного рецепта за другим на одной и той же установке SAP Commerce. Установщик не удаляет рецепты и не восстанавливает исходные настройки файловой системы SAP Commerce. Для установки другого рецепта используйте исходные файлы и каталоги SAP Commerce.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Контроль версий](#)

Следующий:[Расширения](#)

Сопутствующая информация

[Установка SAP Commerce с использованием рецептов установщика](#)

## Установка, сборка и запуск SAP Commerce

Установите SAP Commerce с помощью только\_платформарецепт.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.javale.

```
public void testSuiteIsOnline()
{ assertTrue( canLoginToHybrisCommerce()); }
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testSuiteIsOnline тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testSuiteIsOnline тест
```

### Процедура

.Создайте новый файл рецепта в новом только\_платформапапка называется build.gradle.

```
применить плагин: 'installer-platform-plugin' def
platformOnly = platform {
    после установки {
        обеспечитьAdminPasswordSet()
    }
}
```

```

        }
    настройка задачи {
        doLast {
            platformOnly.setup()
        }
    }
    задача buildSystem(зависит от: настройка) {
        doLast {
            platformOnly.build()
        }
    }
    инициализация задачи (зависит от: buildSystem) {
        doLast {
            platformOnly.initialize()
        }
    }
    начало задачи {
        doLast {
            platformOnly.start()
        }
    }
}

```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать файл самостоятельно или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>\installer\recipes\platform\_only\build.gradlec содержанием <HYBRIS\_HOME\_DIR>/hybris123\src\main\webapp\resources\platform\_only\build.gradle

то же самое \$HYBRIS\_HOME\_DIR/hybris123\src\main\webapp\resources\platform\_only\build.gradle \$HYBRIS\_HOME\_DIR/installer\recipes\platform\_only\build.gr

xcopy /h /y %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\platform\_only\build.gradle %HYBRIS\_HOME\_DIR%\installer\recipes\

.Перейдите к <HYBRIS\_HOME\_DIR>\установщики установить, собрать и запустить SAP Commerce с помощью только платформарецепт.

Это может занять до 20 минут.

```

cd $HYBRIS_HOME_DIR/installer; ./install.sh -r platform_only setup -A local_property:initialpassword.admin=$INITIAL_ADMIN; ./install.sh -r platform_only initialize -A
local_property:initialpassword.admin=$INITIAL_ADMIN;
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh start

```

-Примечание

Если вы получили сообщение об ошибке, сообщающее о том, что невозможно открыть wrapper-macosx-universal-64 или libwrapper-macosx-universal-64.jnilib, поскольку не удалось проверить разработчика, выполните следующие действия:

а. Перейти в системные настройки безопасности Конфиденциальность и перейдите на вкладку Общие.

б. В списке выберите исполняемый файл, который вы пытались запустить, и разблокируйте его.

в) Повторите команду.

```

настройка rem и инициализация hybris
cd %HYBRIS_HOME_DIR%\installer & install.bat -r platform_only setup -A local_property:initialpassword.admin=%INITIAL_ADMIN% & install.bat -r pl rem настроить tomcat как службу Windows
set _YWRAPPER_CONF=%HYBRIS_HOME_DIR%\hybris\bin\platform\tomcat\conf\wrapper.conf cd
%HYBRIS_HOME_DIR%\hybris\bin\platform\tomcat\bin & InstallTomCatService.bat rem start hybris

cd %HYBRIS_HOME_DIR%\hybris\bin\platform и запустить hybrisserver.bat

```

.Дождитесь запуска сервера, а затем откройте бэкэнд SAP Commerce на https://localhost:9002

Это открывает SAP Commerce Administration Console (Administration Console), корневую веб-страницу в SAP Commerce. Вы можете войти в систему и исследовать Administration Console, используя имя пользователя по умолчанию (админ) и пароль, который вы определили как переменную среды.

You're Administrator [logout](#)

Type here...

Platform Monitoring Maintenance Console YHac Extension

## Uptime: 4 minutes. Enjoy!

Show last: 12 hours [refresh](#)

### Memory overview

Committed Memory (blue), Maximum Memory (grey), Used Memory (white)

Time	Committed Memory	Maximum Memory	Used Memory
18:49:05	~1000	~1500	~800
18:51:10	~1050	~1550	~850
18:53:09	~1000	~1500	~800

### CPU Load (4 processors)

CPU Workload

Time	CPU Workload
18:49:05	~80
18:51:10	~70
18:53:09	~75

### Threads overview

Threads

Time	Threads
18:49:05	~100
18:51:10	~100
18:53:09	~100

### Db Connections Overview

Db Connections

Time	Db Connections
18:49:05	~2.0
18:51:10	~2.0
18:53:09	~2.0

**Top links**

- [ImpEx Import](#)
- [FlexibleSearch](#)
- [Home](#)
- [ImpEx Import](#)
- [Database](#)

[Clear history](#)

**Statistics**

- The Charts show a limited number of datasets.
- In the top right you can choose how long the history should reach into the past.
- At 5 min, 1 hour and 12 hours, data polling is activated. At higher time-periods you have to update the values with the refresh button.
- To get a more detailed view, you can mark an area in the chart you want to see. In the upcoming chart, more detailed data in this time-period is shown.

.Запустите `testSuiteIsOnline` повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testSuiteIsOnline тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testSuiteIsOnline тест
```

.Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add .; git commit -m "Установка, сборка и запуск Hybris"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Установка, сборка и запуск Hybris"
```

## Расширения

SAP Commerce имеет модульную архитектуру, в которой новая бизнес-логика разрабатывается в отдельных функционально-специфичных модулях, называемых расширениями.

Расширение — это инкапсулированная часть функциональности SAP Commerce, которая может содержать бизнес-логику, определения типов, веб-приложение и функциональность конфигурации бэк-офиса. В зависимости от потребностей вашего бизнеса ваше решение будет иметь различное количество расширений, все из которых будут подключены к базовой платформе SAP Commerce через модель внедрения зависимостей Spring.

SAP Commerce поставляется с рядом шаблонов расширений и инструментом на основе Ant (называемым `extgen`) для создания новых расширений на основе этих шаблонов. В этом и последующих разделах вы расширяете SAP Commerce, добавляя и разрабатывая собственное пользовательское расширение.

Для вашего проекта «Концертные туры» вы создаете пользовательское расширение под названием `концертные туры`. Для начала вы создаете расширение по умолчанию из шаблона. На последующих этапах вы добавляете требуемые вам пользовательские функции.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Рецепты установщика](#)

Следующий:[Файл localextensions.xml](#)

Сопутствующая информация

[Концепция расширения](#)

## Создать новое расширение

Создайте новое расширение, чтобы начать разработку собственных функций SAP Commerce.

## Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.javaле.

```
public void testExtensionCreatedOk() {
    assertTrue("Новых констант нет",
        FileHelper.fileExists("../hybris/bin/custom/concerttours/src/concerttours/constants/ConcerttoursConstants.java")); assertTrue("Новых услуг нет",
        FileHelper.fileExists("../hybris/bin/custom/concerttours/src/concerttours/service/ConcerttoursService.java")); assertTrue("Новых служб по умолчанию нет",
        FileHelper.fileExists("../hybris/bin/custom/concerttours/src/concerttours/service/impl/DefaultConcerttoursService.java")); assertTrue("Новой настройки нет",
        FileHelper.fileExists("../hybris/bin/custom/concerttours/src/concerttours/ConcerttoursStandalone.java"));
}
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testExtensionCreatedOk test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testExtensionCreatedOk test
```

## Процедура

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Перейдите в папку «Платформа».

```
компакт:$HYBRIS_HOME_DIR/hybris/bin/платформа
```

```
компакт:%HYBRIS_HOME_DIR%\hybris\bin\platform
```

. Запустите setantenv команда для настройки инструмента сборки Ant, поставляемого с SAP Commerce.

```
.. ./setantenv.sh
```

```
setantenv.bat
```

. Выполнить задачу Ant extgen для создания нового пустого расширения SAP Commerce с именем концертные туры и пакета hybris.platform.concerttours.

```
ant extgen -Dinput.template="yempty" -Dinput.name="concerttours" -Dinput.package="concerttours"
```

```
ant extgen -Dinput.template="yempty" -Dinput.name="concerttours" -Dinput.package="concerttours"
```

. Игнорируйте шаги, которые команда ant затем выводит в консоль (например, [echo] 1) Добавьте свое расширение...), поскольку они более подробно изложены ниже.

. Запустите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh start
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и запустить hybrisserver.bat
```

. Запустите testExtensionCreatedOk повторите приемочное испытание и подтвердите, что оно теперь прошло.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testExtensionCreatedOk test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testExtensionCreatedOk test
```

. Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add .; git commit -m "Создать новое расширение"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Создать новое расширение"
```

## Файл localextensions.xml

The localextensions.xml file содержит список расширений, которые ваша конкретная конфигурация SAP Commerce включает во время компиляции и выполнения.

Список расширений по умолчанию, найденных в этом файле, определяется рецептом установщика, который вы используете. Минимальный только платформа Рецепт содержит самый короткий список расширений. Другие рецепты, такие как b2b\_скоритель, содержат гораздо более длинные списки, обеспечивая более богатую функциональность.

Чтобы расширить бизнес-функциональность SAP Commerce, добавьте другие расширения в этот список. Вы можете добавить комбинацию расширений, предоставляемых SAP Commerce, и ваши собственные пользовательские расширения.

Когда вы впервые создаете SAP Commerce, localextensions.xml file содержит только необходимые расширения. Когда вы решите, какие расширения вам нужны или вы хотите использовать, добавьте их в этот файл. В этой процедуре вы уведомляете SAP Commerce о новом концертные туры расширение, добавив его в localextensions.xml file.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Расширения](#)

Следующий:[Модели данных](#)

## Сопутствующая информация

[Установка на основе указанных расширений](#)

### Добавьте свое расширение

Добавьте концептные туры расширение klocalextensions.xml и перестроить SAP Commerce.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.java.

```
public void loginAndCheckForConcertToursExtension()
{
    canLoginToHybrisCommerce();
    navigationTo("https://localhost:9002/platform/extensions");
    assertTrue(waitForExtensionListing("concerttours"));
}
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#loginAndCheckForConcertToursExtension test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#loginAndCheckForConcertToursExtension test
```

### Процедура

. Перейдите к <*\$HYBRIS\_HOME\_DIR*>/hybris/config/localeextensions.xml и добавьте <расширение dir="..<custom/concerttours">> влево.

```
<hybrisconfig xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance' xsi:noNamespaceSchemaLocation='..<bin>/platform/resources/schemas/extensions
<расширения>
<path dir='${HYBRIS_BIN_DIR}' autoload='false' />

<!-- Ваше новое расширение --> <extension dir="..<custom/concerttours">/>

</расширения>
</hybrisconfig>
```

Интерактивный ярлык SAP Commerce 123: если вам не удается обновить localeextensions.xml себя или вы хотите пропустить этот шаг, замените <*\$HYBRIS\_HOME\_DIR*>/hybris/config/localeextensions.xmlc содержанием <*\$HYBRIS\_HOME\_DIR*>/hybris123/src/main/webapp/resources/localeextensions.xml

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/localeextensions.xml $HYBRIS_HOME_DIR/hybris/config/localeextensions.xml
```

```
копировать /у %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\localeextensions.xml %HYBRIS_HOME_DIR%\hybris\config\localeextensions.xml
```

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Перестройте SAP Commerce с помощью Ant.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ant очистить все
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и очистите все
```

. Запустите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh start
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и запустить hybrisserver.bat
```

. Войдите в SAP Commerce Administration Console (Administration Console) по адресу <https://localhost:9002> используя логин, **админ** и пароль, который вы определили как переменную среды.

. Перейдите к **Расширения**, **платформы** и **модули**, передвигаясь прямо к <https://localhost:9002/platform/extensions>.

Здесь вы увидите обзор доступных расширений, а также информацию о том, включают ли какие-либо из этих расширений основной модуль расширения, веб-модуль и/или основной модуль + модуль расширения.

You're Administrator [logout](#)

Platform Monitoring Maintenance Console YHac Extension

## Extensions

Show 50 entries Search:

Name	Version	Core	HMC	Web
advancedsavedquery	6.3.0.0-SNAPSHOT	✓	✗	
catalog	6.3.0.0-SNAPSHOT	✓	✗	
comments	6.3.0.0-SNAPSHOT	✓	✗	
commons	6.3.0.0-SNAPSHOT	✓	✗	
concerttours	6.3.0.0-SNAPSHOT	✓	✗	/concerttours
core	6.3.0.0-SNAPSHOT	✓	✗	
deliveryzone	6.3.0.0-SNAPSHOT	✓	✗	

.Запустите `loginAndCheckForConcertToursExtension` повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#loginAndCheckForConcertToursExtension test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#loginAndCheckForConcertToursExtension test
```

.Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add .; git commit -m "Добавить свое расширение"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Добавьте свое расширение"
```

## Модели данных

Модель данных, лежащая в основе SAP Commerce, определяется в XML-файлах. Новые типы данных для расширений, называемые типами элементов, определяются в `<имя-расширения>-items.xml`.

Новые функции, которые вы добавляете в SAP Commerce, могут потребовать дополнений к модели данных SAP Commerce. Например, вам нужны элементы групп, туров и концептов для моделирования бизнес-требований компании Little Concert.

Thecore-items.xml содержит основные определения наиболее фундаментальных типов элементов и отношений, предоставляемых и используемых платформой. Они включают определения для различных типов пользователей, заказов и продуктов, среди прочих сущностей.

Если расширение добавляет новые типы элементов, расширяет существующие типы элементов, изменяет отношения или добавляет отношения к общей модели данных, оно должно делать это в файле с именем `<имя-расширения>-elements.xml`, расположенным в собственном каталоге ресурсов верхнего уровня.

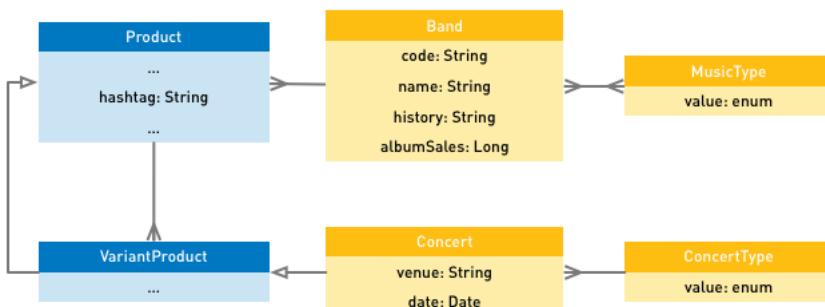
SAP Commerce называет типы данных itemtypes, каждый из которых определен в XML-элементе itemtype. Вы определяете новый тип элемента, добавляя новый itemType-элемент к `<имя-расширения>-elements.xml`. Аналогично, SAP Commerce называет отношения «один ко многим» и «многие ко многим» между типами элементов отношением, и вы определяете новые отношения, добавляя XML-элемент отношения к `<имя-расширения>-elements.xml`.

Во время сборки и инициализации базы данных платформа объединяет все XML-декларации из используемых расширений и генерирует классы Java и схему базы данных.

Для интернет-магазина небольших концертных туров вы представляете каждый концертный тур как базовый продукт, а каждый концерт в рамках тура как вариант базового продукта. Вы также расширяете основную модель данных для обработки нескольких дополнительных требований:

- Каждый концертный тур должен быть связан с хештегом для целей маркетинга в социальных сетях, поэтому для этого вам необходимо добавить строковый атрибут к типу элемента «Продукт».
- Для каждого концерта необходимо указать место и дату проведения, поэтому вы создадите подтип, который расширяет тип элемента VariantProduct с помощью этих атрибутов.
- каждый концерт может проводиться как на открытом воздухе, так и в помещении, поэтому вы создадите список, который сможете использовать для обозначения этого
- Группы, дающие концерты, должны быть представлены в модели данных, поэтому для этого вы определите новый тип элемента Band и новую связь между типами элементов Product и Band.
- вы хотите пометить группы по типам музыки, которую они играют, поэтому вы создадите еще одно перечисление, чтобы обеспечить это

Следующая диаграмма демонстрирует новую модель данных. Желтые элементы — это то, что вам нужно добавить.



## Примечание

Здесь вы не определяете никаких значений для MusicType перечисление, потому что вы будете делать это при инициализации работающей системы. Если вы определили некоторые значения для MusicType здесь, то вы застрянете с ними во время выполнения, потому что, хотя вы можете удалить любые новые значения, добавленные во время выполнения, вы не можете удалить значения, которые определены в \*-элементы.xml лес.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[файл localextensions.xml](#)

Следующий:[Проектирование баз данных](#)

## Сопутствующая информация

[Моделирование продуктов и данных](#)

## Расширить модель данных

Расширьте модель данных SAP Commerce, объявив новые типы данных и связи, а также расширяв типы данных, объявленные другими расширениями.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.java на.

```
public void testExtensionModelOk() выдает ClassNotFoundException, IOException {
    assertTrue("ProductModel не был расширен для поддержки Hashtag и Band",
        FileHelper.fileContains("../hybris/bin/platform/bootstrap/gensrc/de/hybris/platform/core/model/product/ProductModel.java", "getHashtag",
            "получитьБанд",
            "установитьХэштег", "setBand"));

    assertTrue("Новая BandModel не поддерживает Code, Name, History, AlbumSales",
        FileHelper.fileContains("../hybris/bin/platform/bootstrap/gensrc/concerttours/model/BandModel.java", "getName", "getHistory", "getCode",
            "getAlbumSales",
            "setName", "setHistory", "setCode",
            "setAlbumSales"));

    assertTrue("Новый ConcertModel не расширяет VariantProductModel или не поддерживает Venue и Date",
        FileHelper.fileContains("ConcertModel", "../hybris/bin/platform/bootstrap/gensrc/concerttours/model/ConcertModel.java", "ВариантПродуктаМодель",
            расширяет
            "получитьМесто", "получитьДата",
            "установитьМесто", "установитьДата"));

    assertTrue("Новая группа не расширяет GenericItem или не поддерживает Code, Name, History, AlbumSales",
        FileHelper.isBandCreated());

    assertTrue("Новый Concert не расширяет VariantProduct или не поддерживает Venue, Date",
        FileHelper.isConcertCreated());
}
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testExtensionModelOk тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testExtensionModelOk тест
```

### Процедура

. Обновите модель данных в <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttours-items.xml/ле.

а. В пределах типы элементов теги, добавьте новый атрибут к существующему типу, убедившись, что здесь атрибут autoscreate установлен в false, поскольку вы добавляете к itemtype, который определен в другом месте.

```
<itemtype generate="true" code="Product" autocreate="false">
    <атрибуты>
        <атрибут квалификатор="хэштег" тип="java.lang.String">
            <description>хэштег концертного тура для социальных сетей</description> <persistence
                type="property" />
        </атрибут>
    </атрибуты>
</тип_элемента>
```

б. В пределах типы элементов теги, создайте новый тип, убедившись, что атрибут autoscreate установлен в значение true, поскольку вы создаете новый тип элемента.

```
<itemtype generate="true" code="Band" autocreate="true">
    <deployment table="Bands" typecode="30268" /> <attributes>

        <квалификатор атрибута="код" тип="java.lang.String">
            <description>короткий уникальный код диапазона</description>
            <persistence type="property" />
        </атрибут>
        <квалификатор атрибута="имя" тип="java.lang.String">
            <description>название группы</description>
            <persistence type="property" />
        </атрибут>
        <квалификатор атрибута="history" тип="java.lang.String">
            <description>история группы</description>
```

```

        <настойчивость тип="свойство" />
    </атрибут>
    <attribute qualifier="albumSales" тип="java.lang.Long">
        <description>официальные мировые продажи альбома</description>
        <persistence type="property" />
    </атрибут>
</атрибуты>
</тип_элемента>

```

с. В верхней части раздела, определите два новых перечисления.

```

<перечисления>
    <enumtype code="ConcertType" autocreate="true" generate="true" dynamic="false">
        <значение код="openair" />
        <значение код="внутренний" />
    </enumtype>

    <enumtype code="MusicType" autocreate="true" generate="true" dynamic="true"> </enumtype>
</enumtypes>

```

г. В пределах типов элементов, создайте новый тип, расширив другой.

```

<itemtype generate="true" code="Concert" extends="VariantProduct" autocreate="true">
    <атрибуты>
        <квалификатор атрибута="место проведения" тип="java.lang.String">
            <description>место проведения концерта</description>
            <persistence type="property" />
        </атрибут>
        <квалификатор атрибута="дата" тип="java.util.Date">
            <description>дата концерта</description> <persistence
            type="property" />
        </атрибут>
        <attribute qualifier="concertType" type="ConcertType">
            <description>тип концерта (в помещении или на открытом воздухе)</description>
            <persistence type="property" />
        </атрибут>
    </атрибуты>
</тип_элемента>

```

е. Под enumtypes раздел, создайте два новых отношения.

```

<отношения>
    <relation code="Product2RockBand" autocreate="true" generate="false" localized="false">
        <sourceElement qualifier="tours" type="Product" collectiontype="set" cardinality="many" ordered="false">
            <modifiers read="true" write="true" search="true" Optional="true" /> </sourceElement>

        <targetElement qualifier="band" type="Band" cardinality="one">
            <modifiers read="true" write="true" search="true" необязательно="true" /> </targetElement>
        </отношение>
    <relation code="Band2MusicType" autocreate="true" generate="false" localized="false">
        <таблица развертывания="Band2MusicType" typecode="30269" />
        <sourceElement qualifier="bands" type="Band" collectiontype="set" cardinality="many" ordered="false">
            <modifiers read="true" write="true" search="true" Optional="true" /> </sourceElement>

        <targetElement qualifier="types" type="MusicType" cardinality="many">
            <modifiers read="true" write="true" search="true" необязательно="true" /> </targetElement>
        </отношение>
    </отношения>

```

Окончательный файл будет выглядеть примерно так: concerttours-items.xml.

```

<?xml версия="1.0" кодировка="ISO-8859-1"?>
<!- ~ [у] Платформа hybris ~ Авторские права (c) 2000-2016 SAP SE ~ Все права
зарезервировано. ~ ~ Это программное обеспечение является конфиденциальной и частной
информацией SAP ~ Hybris («Конфиденциальная информация»). Вы не должны разглашать такую
~ Конфиденциальную информацию и должны использовать ее только в соответствии с ~
условиями лицензионного соглашения, которое вы заключили с SAP Hybris. --> <!- ВНИМАНИЕ: Это
всего лишь пример файла. Вам необходимо отредактировать его в соответствии с
в соответствии с вашими потребностями. -->

<items xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="items.xsd">

    <!-- Hybris123SnippetStart concerttours-items.xml_enum --> <enumtypes>

        <enumtype code="ConcertType" autocreate="true" generate="true" dynamic="false">
            <значение Код="openair" />
            <значение Код="внутренний" />
        </enumtype>

        <enumtype code="MusicType" autocreate="true" generate="true" dynamic="true"> </enumtype>
    </enumtypes>
    <!-- Hybris123SnippetEnd -->

    <!-- Hybris123SnippetStart concerttours-items.xml_relations --> <relations>

        <relation code="Product2RockBand" autocreate="true" generate="false" localized="false">
            <sourceElement qualifier="tours" type="Product" collectiontype="set" cardinality="many" ordered="false">
                <modifiers read="true" write="true" search="true" Optional="true" /> </sourceElement>

            <targetElement qualifier="band" type="Band" cardinality="one">
                <modifiers read="true" write="true" search="true" необязательно="true" /> </targetElement>
            </отношение>
        <relation code="Band2MusicType" autocreate="true" generate="false" localized="false">
            <таблица развертывания="Band2MusicType" typecode="30269" />
            <sourceElement qualifier="bands" type="Band" collectiontype="set" cardinality="many" ordered="false">
                <modifiers read="true" write="true" search="true" Optional="true" /> </sourceElement>

            <targetElement qualifier="types" type="MusicType" cardinality="many">
                <modifiers read="true" write="true" search="true" необязательно="true" /> </targetElement>
            </отношение>
        </отношения>
    </relations>

```

```

<!-- Hybris123SnippetEnd -->
-->

<типы элементов>

<!-- Hybris123SnippetStart concerttours-items.xml_concert -->
<itemtype generate="true" code="Concert" extends="VariantProduct" autocreate="true">
    <атрибуты>
        <квалификатор атрибута="место проведения" тип="java.lang.String">
            <description>место проведения концерта</description>
            <persistence type="property" />
        </атрибут>
        <квалификатор атрибута="дата" тип="java.util.Date">
            <description>дата концерта</description> <persistence
            type="property" />
        </атрибут>
        <attribute qualifier="concertType" type="ConcertType">
            <description>тип концерта (в помещении или на открытом воздухе)</description>
            <persistence тип="свойство" />
        </атрибут>
    </атрибуты>
</тип_элемента>
<!-- Hybris123SnippetEnd -->
-->

<!-- Hybris123SnippetStart concerttours-items.xml_хэштег --> <itemtype
generate="true" code="Product" autocreate="false">
    <атрибуты>
        <атрибут квалификатор="хэштег" тип="java.lang.String">
            <description>хэштег концертного тура для социальных сетей</description> <persistence
            type="property" />
        </атрибут>
    </атрибуты>
</тип_элемента>
<!-- Hybris123SnippetEnd -->
-->

<!-- Hybris123SnippetStart concerttours-items.xml_Band --> <itemtype
generate="true" code="Band" autocreate="true">
    <deployment table="Bands" typecode="30268" /> <attributes>

        <квалификатор атрибута="код" тип="java.lang.String">
            <description>короткий уникальный код диапазона</description>
            <persistence type="property" />
        </атрибут>
        <квалификатор атрибута="имя" тип="java.lang.String">
            <description>название группы</description>
            <persistence type="property" />
        </атрибут>
        <квалификатор атрибута="history" тип="java.lang.String">
            <description>история группы</description> <persistence
            type="property" />
        </атрибут>
        <attribute qualifier="albumSales" тип="java.lang.Long">
            <description>официальные мировые продажи альбомов</description>
            <persistence тип="свойство" />
        </атрибут>
    </атрибуты>
</тип_элемента>
<!-- Hybris123SnippetEnd -->
-->

```

&lt;/itemtypes&gt;

&lt;/элементы&gt;

Интерактивный ярлык SAP Commerce 123: если вы не можете обновить модель данных самостоятельно или хотите пропустить эти шаги, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttours-items.xmlc содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours-items.xml

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-items.xml $HYBRIS_HOME_DIR/hybris/bin/custom/concer
```

```
копировать /у %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\resources\concerttours-items.xml %HYBRIS_HOME_DIR%\hybris\bin\custom
```

.Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

.Перестройте SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ant очистить все
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и очистите все
```

Темуравей очистить всекоманда гарантирует, что вы сначала удалите все ранее созданные классы Java, включая те, которые больше не нужны.

.Запустите testExtensionModelOk повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testExtensionModelOk тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testExtensionModelOk тест
```

.Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add . ; git commit -m "Расширить модель данных"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Расширить модель данных"
```

## Проектирование баз данных

SAP Commerce хранит свои данные в реляционной базе данных. Поддерживается большой спектр сторонних баз данных, что позволяет вам выбрать базу данных, наиболее подходящую для вашего решения.

При определении новой модели данных необходимо указать, как эта модель должна сохраняться в базе данных. В частности, следует ли сохранять каждый новый тип элемента в его собственной таблице базы данных или в родительской таблице. При определении новых типов элементов можно включить атрибут развертывания XML, чтобы указать имя новой таблицы, специфичной для этого типа. Если не включить атрибут развертывания XML в новый тип элемента, SAP Commerce сохранит этот элемент в таблице, используемой родительским элементом нового типа.

Например, если вы не указали тег развертывания для Концепттипа товара, SAP Commerce сохраняет товары этого типа в той же таблице базы данных, что и товары ВариантПродукттипа элемента, потому что вы определили Концепткак расширение ВариантПродукта. The ВариантПродукттип также не имеет предложения по развертыванию, но он расширяет Продукттип элемента, который имеет пункт развертывания. Это означает, что элементы Концепт, ВариантПродукт, и Продукт хранятся в одной и той же таблице базы данных, которая называется Продукция, как указано в Продукт определение типа элемента всогоре-items.xml. Это, как правило, то, что вам нужно, потому что это упрощает поиск и извлечение подмножеств всех видов продуктов. Не требуется никаких объединений таблиц базы данных.

Напротив, вы не определили Группатип элемента как расширение любого другого типа элемента. Поэтому элементы нашего Группатип элемента будут сохранены в таблице, указанной корневым типом элемента, который называется GenericItem. Все типы расширяются GenericItemесли не указано иное. Это почти наверняка не то, что вам нужно, поэтому, когда вы специально не расширяете другой тип элемента, вам всегда следует включать предложение развертывания.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Модели данных](#)

Следующий:[Атрибуты](#)

## Сопутствующая информация

[Система типов](#)

[Сторонние базы данных](#)

## Обновить базу данных

Инициируйте обновление системы, чтобы отразить изменения, внесенные вами в модель данных, и подтвердите, что вы создали новую группастол.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест Hybris123Tests.java.

```
public void testDatabaseSetup() выдает исключение {
    HsqlDBHelper hsqlDb = new HsqlDBHelper(); // Примечание: тест завершится неудачей, если набор одновременно запущен на этой базе данных. try {

        String res = hsqlDb.select("ВЫБЕРИТЕ ИМЯ_ТАБЛИЦЫ ИЗ ИНФОРМАЦИОННОЙ_СХЕМЫ.SYSTEM_COLUMNS, ГДЕ ИМЯ_ТАБЛИЦЫ НЕ ПОДОБНО 'SYSTEM_%'");
        assertTrue("Не удалось найти таблицу BANDS", res.contains("BANDS"));
    }
    поймать (Исключение e) {
        fail("testDatabaseSetup", "HsqlDBTest не пройден: " + e.getMessage());
    }
    окончательно {
        hsqlDb.shutdown();
    }
}
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testDatabaseSetup тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testDatabaseSetup тест
```

### Процедура

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Обновите базу данных.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ant обновление системы
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и ant updatesystem
```

. Откройте базу данных с помощью hsqlDb DatabaseManager по умолчанию.

```
cd $HYBRIS_HOME_DIR/hybris; java -cp ./bin/platform/lib/dbdriver/hsqlDb*.jar org.hsqlDb.util.DatabaseManager --url jdbc:hsqlDb:file:/data/hsql
```

```
cd %HYBRIS_HOME_DIR%\hybris и start /B java -cp .\bin\platform\lib\dbdriver\* org.hsqlDb.util.DatabaseManager --url jdbc:hsqlDb:file:\data\hsq
```

Должна быть новая таблица под названием Голосы.

-Кончик

Когда Hybris запущен, вы также можете просматривать базу данных в SAP Commerce Administration Cockpit по адресу <https://localhost:9002> передя **Размер таблицы** базы данных мониторинга.

➤ Рассчитать размеры таблицы ищут в списке таблиц, найденных в базе данных.

. Закройте менеджер баз данных hsqldb.

`pskill -f "Менеджер баз данных"`

`taskkill /FI "ЗАГОЛОВОК ОКНА экв HSQL*"`

. Запустите `testDatabaseSetup` повторите приемочное испытание и подтвердите, что оно теперь пройдено.

`cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testDatabaseSetup тест`

`cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testDatabaseSetup тест`

. Зафиксируйте изменения в локальном репозитории Git.

`cd $HYBRIS_HOME_DIR/hybris; git add . ; git commit -m "Обновить базу данных"`

`cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Обновить базу данных"`

## ИмпЭкс

SAP Commerce поставляется с мощными функциями импорта и экспорта текстовых данных под названием ImpEx.

Файлы ImpEx по сути являются файлами, разделенными запятыми (CSV), которые позволяют осуществлять компактный, понятный человеку импорт и экспорт данных в SAP Commerce и из него. Их можно вручную запускать через SAP Commerce Administration Console или автоматически запускать каждый раз при инициализации системы, сохраняя файл ImpEx в соответствии с простым соглашением и в определенном месте.

Консоль администрирования SAP Commerce предоставляет интерфейс, в [Импекс Импортвкладка Консоль](#), с помощью которого вы можете вручную импортировать небольшие объемы данных ImpEx.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Проектирование баз данных](#)

Следующий:[Основные и проектные данные по соглашению](#)

## Загрузка данных через консоль администрирования

Импортируйте данные Band и Group с помощью консоли администрирования SAP Commerce.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест `Hybris123Tests.java`.

```
public void testManualImpex() выдает исключение
{ canLoginToHybrisCommerce();
navigationTo("https://localhost:9002/console/flexsearch"); waitForFlexQueryFieldThenSubmit("SELECT {pk},
{code}, {history} FROM {Band}"); assertTrue(waitFor("td", "Группа пения а капелла из Мюнхена"));

}
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

`cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testManualImpex test`

`cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testManualImpex test`

### Процедура

. Запустите SAP Commerce.

`cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh start`

`cd %HYBRIS_HOME_DIR%\hybris\bin\platform` и запустить `hybrisserver.bat`

. Импортируйте несколько групп и диапазонов.

а. Войдите в консоль администрирования SAP Commerce по адресу <https://localhost:9002> используя логин `admin` и пароль, который вы определили как переменную среды.

б. Перейти к Консольвкладку и выберите [Импекс Импорт](#) вариант.

The [Импекс Импорт](#) является страница.

в. Скопируйте и вставьте следующий текст в [Импортировать контент](#) текстовое поле.

# ImpEx для импорта групп в магазин Little Concert Tours

```
INSERT_UPDATE Группа; код[unique=true]; название; альбом Продажи; история
;A001;уRock;1000000;Нерегулярная трибют-рок-группа, в состав которой входят старшие менеджеры ведущего поставщика коммерческого программного обеспечения ;A006;уBand; ;Голландская трибют-рок-группа, образованная в 2013 году, исполняющая классические рок-мелодии шестидесятых, семидесятых и восьмидесятых годов ;A003;уJazz;7;Экспериментальная джазовая группа из Лондона, исполняющая множество музыкальных нот одновременно в неожиданных комбинациях и последовательностях
```

;A004;Запрещено;427;Возрожденный польский бой-бэнд 1990-х годов — этот жанр поп-музыки в его самой сомнительной лучшей форме ;A002;Sirken;2000;Группа пения а капелла из Мюнхена; вдохновляющая смесь традиционных и современных песен ;A005;The Choir;49000;Восторженный, шумный госпел-хор, исполняющий традиционные госпел-песни с юга ;A007;The Quiet;1200;Английское хоровое общество, специализирующееся на прекрасно аранжированных, успокаивающих мелодиях и песнях

г. Нажмите кнопку Проверить содержание кнопку для выполнения базовой проверки синтаксиса содержимого.

Сообщение появляется выше Импортировать контент кнопка, подтверждающее, что содержимое является действительным.

е. Нажмите кнопку Импортировать контент кнопку для запуска импорта.

Интерактивный ярлык SAP Commerce 123: если вы не можете импортировать данные полос и групп самостоятельно или хотите пропустить этот шаг, выполните следующую команду.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#simulateManualImpex test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#simulateManualImpex test
```

. Подтвердите, что импорт прошел успешно.

а. В консоли администрирования SAP Commerce перейдите в Консоль кладку и выберите Гибкий Поиск вариант.

The Гибкий Поиск появляется страница.

б) Извлеките все элементы данных Band из базы данных, введя запросы выберите {pk},{code},{history} из {Band} в Гибкий Поиск текстовое поле запроса и щелкните Выполнить кнопка.

-Кончик

Вы также можете нажать на Все продукты ссылка в Образцы запросов поле справа на странице и замените имя [de] атрибут систория Продукт Группа содержанием запроса, которые появляются.

с. Проверьте, что все импортированные элементы данных Band присутствуют и верны в Результаты поиска кладка.

Элементы данных Band отображаются в результатах поиска с атрибутом PK (первичный ключ), который добавляется для каждого элемента механизмом сохранения SAP Commerce при импорте.

. Запустите testManualImpex повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testManualImpex test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testManualImpex test
```

. Задокументируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add .; git commit -m "Загрузить данные через HAC"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Загрузить данные через HAC"
```

## Основные и проектные данные по соглашению

Вы можете подготовить файлы ImpEx, содержащие основные и проектные данные, которые система импортирует при инициализации. Эта функциональность следует принципу Convention over Configuration (CoC).

Использование ImpEx для импорта данных через SAP Commerce Administration Console (Administration Console) полезно для adhoc-импорта, но не для многократной загрузки больших объемов данных ImpEx. Содержимое каждый раз при инициализации или обновлении базы данных. Чтобы автоматически загружать эти данные, вы можете поместить свой ImpEx в текстовые файлы в /resources/impexp каталога вашего расширения. Соглашение об именовании затем гарантирует, что SAP Commerce автоматически загрузит их как часть процесса инициализации или обновления.

Файлы ImpEx делятся на две категории: те, которые описывают основные данные, и те, которые описывают данные о проекте или продукте.

### Основные данные ImpEx le

Содержит основные справочные данные, которые требуются для вашего расширения. Essential data всегда импортируются при инициализации платформы с вашим расширением. Essential data ImpEx файлы имеют имена в формате essentialdata-\*.impeks.

### Данные проекта ImpEx le

Содержит данные, которые являются необязательными для вашего расширения, например, данные образца. Данные проекта включаются только при установке флагка данных проекта для вашего расширения в консоли администрирования во время инициализации. Файлы ImpEx данных проекта имеют имена в формате projectdata-\*.impeks.

Если найдено несколько файлов, соответствующих соглашению об именовании, SAP Commerce загружает и обрабатывает их в произвольном порядке. Если порядок важен, одним из простых решений является создание одного файла, соответствующего соглашению об именовании. Затем вы используете функцию включения ImpEx, чтобы включить содержимое из других файлов в требуемом порядке, и переименовываете включенные файлы так, чтобы они больше не соответствовали соглашению об именовании.

В этом примере файлы ImpEx намеренно созданы для разделения всего контента для каждого тура в отдельный файл и использования ВСТАВИТЬ\_ОБНОВЛЕНИЕ команды. Поэтому, если нужно импортировать много туров и концертов, то порядок обработки файлов не важен.

Родительская тема: [Изучите тип SAP Commerce](#)

Предыдущий: [ИмпЭкс](#)

Следующий: [Основные и проектные данные по коду](#)

Сопутствующая информация

[Импорт ImpEx для основных и проектных данных](#)

## Импортируйте важные данные в свою модель данных

Используйте ImpEx для настройки основных и проектных данных для вашего расширения.

## Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.java.

```
public void testCoCImpex() выдает исключение
{
    canLoginToHybrisCommerce();
    navigationTo("https://localhost:9002/platform/init");
    waitForThenClickButtonWithText("Инициализировать");
    waitForThenClickOkInAlertWindow();
    waitForInitToComplete();
    закрытьБраузер();

    canLoginToHybrisCommerce(); navigationTo("https://localhost:9002/console/flexsearch");
    waitForFlexQueryFieldThenSubmit("SELECT {pk}, {code}, {history} FROM {Band}");
    assertTrue(waitFor("td", "Группа пения а капелла из Мюнхена"));

}
}
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testCoCImpex тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testCoCImpex тест
```

## Процедура

. Создайте essentialdata-bands.impex в *<HYBRIS\_BIN\_DIR>/custom/concerttours/recursys/impex*

а) Создайте файл ImpEx с основными данными.

```
# ImpEx для импорта групп в магазин Little Concert Tours
```

```
INSERT_UPDATE Группа; код[unique=true]; название; альбомПродажи; история
;A001;yRock;1000000;Нерегулярная трибют-рок-группа, состоящая из старших менеджеров ведущего поставщика коммерческого программного обеспечения ;A006;yBand;;Голландская трибют-рок-группа, образованная в 2013 году, исполняющая классические рок-мелодии шестидесятых, семидесятых и восьмидесятых годов ;A003;jazz;7;Экспериментальная джазовая группа из Лондона, играющая множество музыкальных нот вместе в неожиданных комбинациях и последовательностях ;A004;запрещено;427;Возрожденныйпольский бой-бэнд из 1990-х годов — этот жанр поп-музыки в своем самом сомнительном проявлении ;A002;Sirkens;2000;Группа пения а капелла из Мюнхена; вдохновляющая смесь традиционных и современных песен ;A005;Хор;49000;Восторженный, шумный госпел-хор, исполняющий традиционные госпел-песни с далекого юга ;A007;The Quiet;1200;Английское хоровое общество, специализирующееся на прекрасно аранжированных, успокаивающих мелодиях и песнях
```

б) Создайте файл ImpEx данных проекта.

```
# ImpEx для импортного тура и дат для группы
```

```
# Макросы / Замена параметров определения $productCatalog=concertToursProductCatalog $superCategories=superCategories(code, $catalogVersion) $baseProduct=baseProduct(code, $catalogVersion) $approved=approvalstatus(code)[default='approved'] $catalogVersion=catalogversion(catalog(id[default=$productCatalog]),version[default='Online']) [unique=true,default=$productCatalog;Online]
```

```
# Каталог продукции
INSERT_UPDATE Каталог; id[unique=true];
$productCatalog
```

```
# Версия каталога продукта
INSERT_UPDATE CatalogVersion; catalog(id)[unique=true]; version[unique=true]; active; languages(isoCode); readPrincipals(uid);
$productCatalog; Online; true; en; employeegroup
```

```
# Вставьте продукты
INSERT_UPDATE Продукт; код[ уникальный=истина]; название; группа(код); $суперкатегории; название производителя; идентификатор производителя; единица(код); ЕАН; тип варианта(код ;201701;турне The Grand Little x;A001;x;штук; Концерт
```

```
# Вставьте продукты
INSERT_UPDATE Концерт; код[unique=true]; название; дата[dateformat=dd.MM.yyyy]; место проведения; тип концерта(код); baseProduct(код); $catalogVersion; $approv ;20170101; Гранд-тур - Мюнхен; 03.02.2017; "hybris Мюнхен, Германия"; openair; 201701;
;20170102; Гранд-тур - Лондон; 21.03.2017; "hybris Лондон, Великобритания"; под открытым небом; 201701;
;20170103; Гранд-тур - Монреаль; 15.06.2017; "hybris Монреаль, Канада"; в помещении; 201701; ;20170104; Гранд-тур - Гливице; 04.11.2017; "hybris Гливице, Польша"; в помещении; 201701; ;20170105; Гранд-тур - Боулдер; 07.01.2018; "hybris Боулдер, США"; в помещении; 201701; ;20170106; Гранд-тур - Бостон; "hybris Бостон, США"; 201701;
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать файлы ImpEx самостоятельно или хотите пропустить эти шаги, скопируйте *<HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/impex/\*.impex* в *<HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/impex/*

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/impex/essentialdata-bands.impex \$HYBRIS\_HOME\_DIR/hybris/bin/custom/concerttours/reso

xcopy %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\impex\\*.impex %HYBRIS\_HOME\_DIR%\hybris\bin\custom\concerttours\resources\impex\

. Повторно инициализируйте базу данных с помощью консоли администрирования SAP Commerce.

а. Поместите курсор над Платформа вкладку, чтобы отобразить подменю, а затем щелкните на Инициализация вариант, чтобы перенести вас в <https://localhost:9002/platform/init>

б. В Данные проектов области настроек убедитесь, что все флаги установлены.

в) Нажмите одну из кнопок Инициализировать кнопки.

. Проверьте файл журнала, чтобы убедиться, что оба файла были обработаны без ошибок.

а. Откройте последний файл журнала консоли в папке /hybris/log/tomcat/catalog с помощью текстового редактора или инструмента просмотра файлов журнала.

б. Найдите в файле журнала место, где были обработаны два файла ImpEx, и убедитесь, что при импорте не было зарегистрировано никаких ошибок.

12/3/24, 11:01 утра

.Проверьте, что импорт прошел успешно, используя консоль администрирования FlexibleSearch для выполнения нескольких запросов и подтверждения того, что данные из обоих файлов были импортированы правильно.

.Запустите тест CoCImpex приложением и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testCoCImpex test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testCoCImpex test
```

.Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add . ; git commit -m "Импорт важных данных в вашу модель данных"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Импорт важных данных в вашу модель данных"
```

## Основные и проектные данные по коду

Вы можете подключиться к процессу инициализации и обновления системы, чтобы загрузить данные проекта во время инициализации платформы.

После некоторых размышлений вы решаете, что данные диапазона не являются действительно необходимыми данными и должны быть загружены как данные проекта. Необходимые данные зарезервированы для данных, без которых система не может работать. По этой причине переместите данные диапазона из необходимых данных и настройте их как данные проекта. Хотя результаты не впечатляют, весь ваш код все равно будет работать, если в базе данных не будет сохранено ни одного диапазона.

Поскольку данные вашего проекта могут значительно вырасти и стать более сложными, вы можете применить сложный подход, подключившись к процессу инициализации и обновления.

Ваш простой класс Java использует API уровня сервиса ImpEx для явной загрузки ваших двух файлов ImpEx. Основная работа выполняется вимпексИмпортМетод. Вы настраиваете ИмпортКонфигурацииобъект с различными опциями импорта, включая имя файла, который вы хотите импортировать. Затем вы предоставляем его в качестве параметра для импорта Данныхвонок. Система проверяет полученный ИмпортРезультат на предмет ошибок.

В коде также показаны встроенные в платформу средства ведения журнала.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Основные и проектные данные по соглашению](#)

Следующий:[Основные и проектные данные по конвенции и кодексу](#)

Сопутствующая информация

[Хуки для процесса инициализации и обновления](#)

## Подключитесь к процессу инициализации и обновления

Создайте один или несколько классов Java для подключения к процессу инициализации/обновления и используйте API Impex ServiceLayer для управления данными и их загрузки.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.java на.

```
public void testHookImpex() выдает исключение
{ canLoginToHybrisCommerce();
navigationTo("https://localhost:9002/platform/init");
waitForThenClickButtonWithText("Инициализировать");
waitForThenClickOkInAlertWindow();
waitForInitToComplete();

long timeSinceHookLogsFound = LogHelper.getMSSinceThisWasLogged("Загрузка данных пользовательского проекта для расширения Concerttours завершена"); assertTrue("Не удалось найти ожидаемые журналы "+ timeSinceHookLogsFound,
времяSinceHookLogsFound < 10000); }
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testHookImpex test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testHookImpex test
```

### Процедура

.Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

.Переименовать essentialdata-bands.impex концертные туры-группы.импехи переименовать projectdata-yBandTour.impex концертные туры-yBandTour.impex чтобы гарантировать, что файлы больше не будут загружаться по соглашению.

```
mv $HYBRIS_HOME_DIR/hybris/bin/custom/concerttours/resources/impex/essentialdata-bands.impex $HYBRIS_HOME_DIR/hybris/bin/custom/concerttours/re
```

переместить %HYBRIS\_HOME\_DIR%\hybris\bin\custom\concerttours\resources\impex\essentialdata-bands.impex %HYBRIS\_HOME\_DIR%\hybris\bin\custom\concerttour

. Создайте класс настройки с именем ConcerttoursCustomSetup.java в вашем <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/setup пакете.

Следующий код использует аннотации на уровне класса и метода для подключения класса к процессу инициализации/обновления и для указания того, где в этом процессе должны быть размещены конкретные методы. вызывается. Например, аннотация SystemSetup нарушает MyEssentialData() Метод имеет атрибут типа, указывающий, что метод должен вызываться как часть основной фазы загрузки данных процесса инициализации/обновления.

```
упаковка концертные туры установка;
импорт de.hybris.platform.core.initialization.SystemSetup;
импорт de.hybris.platform.servicelayer.impex.ImportConfig;
импорт de.hybris.platform.servicelayer.impex.ImportResult;
импорт de.hybris.platform.servicelayer.impex.ImportService;
импорт de.hybris.platform.servicelayer.impex.impl.StreamBasedImpExResource; java.io.InputStream;
импорт
импорт org.slf4j.Logger;
импорт org.slf4j.LoggerFactory;

@SystemSetup(extension = "concerttours") открытый
класс ConcerttoursCustomSetup {

    частный статический финальный Logger LOG = LoggerFactory.getLogger(ConcerttoursCustomSetup.class); частный
    ИмпортСервис импортСервис;
    публичный ИмпортСервис getImportService()
    {
        возвращаться импортСервис;
    }
    public void setImportService(final ImportService importService) {

        этот.importService = importService;
    }
    @SystemSetup(type = SystemSetup.Type.ESSENTIAL) public
    boolean putInMyEssentialData()
    {
        LOG.info("Запуск загрузки пользовательских основных данных для расширения Concerttours");
        LOG.info("Загрузка пользовательских основных данных для расширения Concerttours завершена."); return true;
    }
    @SystemSetup(type = SystemSetup.Type.PROJECT) public
    boolean addMyProjectData()
    {
        LOG.info("Начало загрузки данных пользовательского проекта для расширения Concerttours");
        impexImport("/impex/concerttours-bands.impex");
        impexImport("/impex/concerttours-yBandTour.impex");
        LOG.info("Загрузка данных пользовательского проекта для расширения Concerttours завершена."); return
        true;
    }
    защищенный логический impexImport(final String filename) {

        final String message = "Concerttours impexing [" + filename + "]...";
        попробуйте (final InputStream resourceAsStream = getClass().getResourceAsStream(имя_файла)) {

            LOG.info(сообщение);
            окончательный ImportConfig importConfig = new ImportConfig(); importConfig.setScript(new
            StreamBasedImpExResource(resourceAsStream, "UTF-8")); importConfig.setLegacyMode(Boolean.FALSE);

            окончательный ImportResult importResult = getImportService().importData(importConfig); если
            (importResult.isError())
            {
                LOG.error(сообщение + "НЕУДАЧНО");
                вернуть false;
            }
            поймать (finalное исключение e)
            {
                LOG.error(сообщение + "FAILED", e); вернуть
                false;
            }
            вернуть истину;
        }
    }
}
```

Интерактивный ярлык SAP Commerce 123: если вам не удается создать ConcerttoursCustomSetup класс настройки в концертные туры.настройка пакет или вы хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/setup/ConcerttoursCustomSetup.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/setup/ConcerttoursCustomSetup.java

UNIX/Mac

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/setup/ConcerttoursCustomSetup.java

\$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/setup/ConcerttoursCustomSetup.java

Окна

echo f | xcopy %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\setup\ConcerttoursCustomSetup.java %HYBRIS\_H

. Зарегистрируйте этот класс как Spring bean в <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttours-spring.xml

```
<bean id="ConcerttoursCustomSetup" class="concerttours.setup.ConcerttoursCustomSetup">
    <имя_свойства="importService" ref="importService"/> </bean>
```

Интерактивный ярлык SAP Commerce 123: если вы не можете зарегистрировать класс как компонент Spring или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttours-spring.xml содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-spring-withCustomSetup.xml

cp \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-spring-withCustomSetup.xml \$HYBRIS\_HOME\_DIR/hybris/

копировать /у %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\resources\concerttours-spring-withCustomSetup.xml %HYBRIS\_HOME\_DIR%

. Перестройте SAP Commerce с помощью Ant.

cd \$HYBRIS\_HOME\_DIR/hybris/bin/platform; ant очистить все

12/3/24, 11:01 утра

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и очистите все
```

. Запустите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh start
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и запустить hybrisserver.bat
```

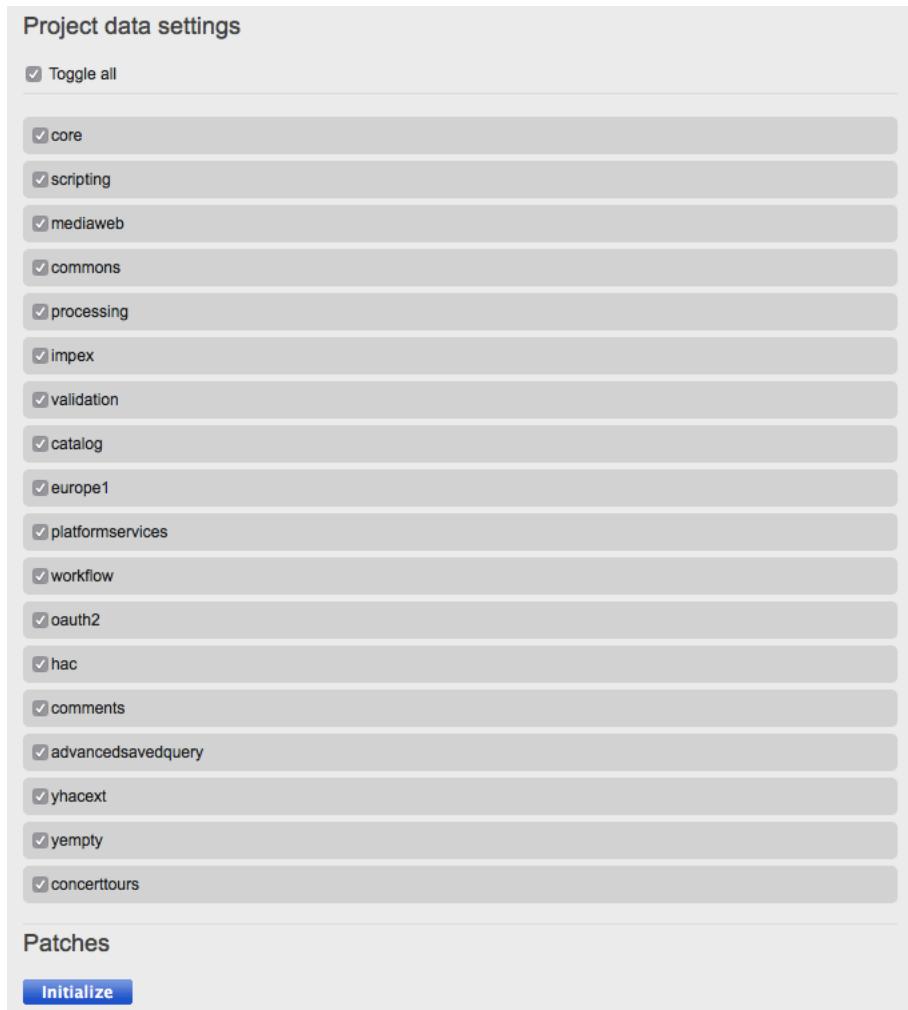
. Повторно инициализируйте консоль администрирования SAP Commerce (консоль администрирования).

а. Войдите в SAP Commerce Administration Console (Administration Console) по адресу <https://localhost:9002> используя логин, админы пароль, который вы определили как переменную среды.

б. Перейдите в [Инициализация платформы](#) или перейдите прямо <https://localhost:9002/platform/init>.

в. В [Настройки данных проекта](#) убедитесь, что все флаги установлены.

г. Нажмите кнопку [Инициализировать](#) кнопка.



. В выводе консоли или в файле журнала консоли в <%HYBRIS\_HOME\_DIR%>/hybris/log/tomcat каталог, найдите вывод, созданный ConcerttoursCustomSetup для подтверждения того, что новый класс был использован.

. Запустите `testHookImpex` повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testHookImpex test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testHookImpex test
```

. Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add .; git commit -m "Подключиться к процессу инициализации и обновления"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Подключиться к процессу инициализации и обновления"
```

## Основные и проектные данные по конвенции и кодексу

SAP Commerce ищет и загружает данные из файлов ImpEx, которые следуют определенному соглашению об именовании. Такое поведение поддерживает парадигму проектирования программного обеспечения «соглашение поверх конфигурации».

Любые файлы ImpEx, которые следуют соглашению об именовании SAP Commerce, автоматически загружаются при инициализации системы. Вы можете использовать это для настройки основных данных, которые требуются для запуска вашего расширения. Вы также можете по желанию предоставить некоторые начальные данные, такие как данные образца или начальные значения в таблице категорий, например, которые система также загружает для удобства. Последний тип данных известен как данные проекта.

Соглашение об именовании для этих двух типов данных следующее:

- `essentialdata-*.*.impexp`

12/3/24, 11:01 утра

- projectdata-\*.impex

В каждом случае \* может быть любым именем по вашему выбору, но принято использовать имя вашего проекта или тип данных.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Основные и проектные данные по коду](#)

Следующий:[Уровень обслуживания](#)

## Используйте соглашение об именовании для загрузки данных

Создайте два файла ImpEx, используя соглашение об именовании SAP Commerce ImpEx, чтобы загрузить данные проекта и добавить теги музыкальных стилей к вашим группам.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.java.

```
public void testHookAndCoC() выдает исключение
{ canLoginToHybrisCommerce();
navigationTo("https://localhost:9002/platform/init");
waitForThenClickButtonWithText("Инициализировать");
waitForThenClickOkInAlertWindow();
waitForInitToComplete();
long timeSinceHookLogsFound = LogHelper.getMSSinceThisWasLogged("импорт ресурса: /impex/projectdata-musictypes.impex"); assertTrue("Не удалось найти
ожидаемые журналы",
времяSinceHookLogsFound < 10000); }
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testHookAndCoC тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testHookAndCoC тест
```

### Процедура

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Создайте два файла ImpEx:projectdata-musictypes.impex и Essentialdata-musictypes.impex в  
<HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/impex/

```
# ImpEx для импорта назначений музыкальных типов Bands
INSERT_UPDATE Band;code[unique=true];types(code) ;A001;rock,eighties
;A006;рок,шестидесятые,семидесятые,восьмидесятые,мужскойвокал ;A003;джаз,женскийвокал
;A004;девяностые,мужскойвокал,поп ;A002;хоровой,поп
;A005;евангелие
;A007;хоровой,классический
```

```
# ImpEx для импорта значений перечисления типов музыки в магазин Little Concert Tours
INSERT_UPDATE MusicType;code[unique=true]
;камень
;джаз
;классический
;поп
;евангелие
;хоровой
;шестидесятые
;семидесятые
;восьмидесятые
;девяностые
;мужскойВокал
;женскийВокал
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать файлы ImpEx или хотите пропустить этот шаг,  
переместите <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/impex/hookwithcoc/\*.impex  
<HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/impex/

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/impex/hookwithcoc/*.impex $HYBRIS_HOME_DIR/hybris/bin/custom/concerttours/resources/imp
```

```
копировать %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\impex\hookwithcoc\*.impex %HYBRIS_HOME_DIR%\hybris\bin\custom\concerttours\resources
```

. Запустите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh start
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и запустить hybrisserver.bat
```

. Повторная инициализация в консоли администрирования SAP Commerce (консоль администрирования).

а. Войдите в SAP Commerce Administration Console (Administration Console) по адресу <https://localhost:9002> используя логин, админ пароль, который вы определили как переменную среды.

б. Перейдите в [Инициализация платформы](#) или перейдите прямо на <https://localhost:9002/platform/init>.

в. В [Настройки данных проекта](#) убедитесь, что все флагки установлены.

г. Нажмите кнопку [Инициализировать](#).

**Project data settings**

**Toggle all**

- core
- scripting
- mediaweb
- commons
- processing
- impex
- validation
- catalog
- europe1
- platformservices
- workflow
- oauth2
- hac
- comments
- advancedsavedquery
- yhacext
- yempty
- concerttours

**Patches**

**Initialize**

. В выводе консоли или в файле журнала консоли в <`HYBRIS_HOME_DIR`>/hybris/log/tomcat каталог, найдите вывод, созданный `ConcerttoursCustomSetup` для подтверждения того, что новый класс был использован.

. Запустите `testHookAndCo` Способите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testHookAndCoC тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testHookAndCoC тест
```

. Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add . ; git commit -m "Использование соглашения об именовании для загрузки данных"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Использовать соглашение об именовании для загрузки данных"
```

## Уровень обслуживания

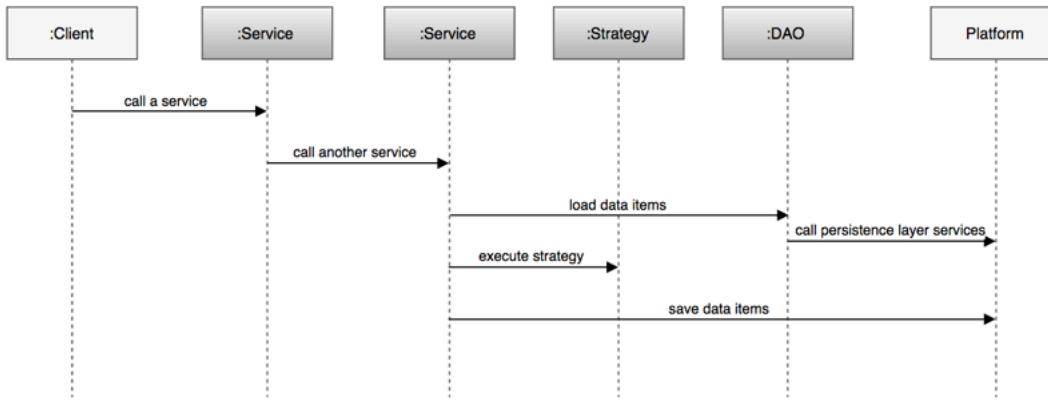
При реализации новой бизнес-логики вы разделяете бизнес-код на классы Java, называемые сервисами. Каждый сервис реализует определенное, четко определенное требование.

Услуги являются частью Уровень обслуживания. Этот слой является логическим уровнем между клиентом и слоем сохранения. Каждая служба имеет свой собственный интерфейс Java, который перечисляет публичные методы, которые могут вызываться клиентами и другими службами.

Служба может содержать всю необходимую бизнес-логику, но чаще всего делегирует ее одному или нескольким из следующих компонентов:

- Другие услуги, обеспечивающие часть требуемого поведения
- Стратегические объекты, которые обеспечивают сменное поведение для различных требований, например, различные алгоритмы расчета корзины
- Объекты доступа к данным, которые обрабатывают поиск и извлечение элементов данных из базы данных.

Эти другие службы, объекты стратегии и объекты доступа к данным также определяются как компоненты Spring с соответствующими интерфейсами и реализациями.



Чтобы создать свой сервис, вы создаете следующие файлы висточникапаконцертные турырасширение:

- BandDAO:Интерфейс DAO, описывающий необходимую вам функциональность CRUD, в данном случае — только функциональность чтения.
- DefaultBandDAO:Реализация DAO
- BandService:Интерфейс обслуживания полосы
- DefaultBandService:Реализация службы полосы

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Основные и проектные данные по конвенции и кодексу](#)

Следующий:[Интеграционные тесты](#)

Сопутствующая информация

[ServiceLayer](#)

## Расширить ServiceLayer

Добавьте новые функциональные возможности к вашему расширению для составления списка и поиска диапазонов, добавив новый интерфейс сервиса, реализацию сервиса и поддержку DAO.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.java.

```

public void testServiceLayerClassesExist() throws IOException {
    // Если вы правильно добавили расширение, то должны появиться новые папки и файлы
    assertEquals("You should have added concerttours.daos.BandDAO.java", FileHelper.fileExistsAndContains(
        "./hybris/bin/custom/concerttours/src/concerttours/daos/BandDAO.java", "public interface BandDAO"));
    assertEquals("Вы должны были добавить concerttours.daos.impl.DefaultBandDAO.java", FileHelper.fileExistsAndContains(
        "./hybris/bin/custom/concerttours/src/concerttours/daos/impl/DefaultBandDAO.java", "public class DefaultBandDAO implements BandDAO"));
    assertEquals("Вы должны были изменить concerttours-spring.xml", FileHelper.fileExistsAndContains(
        "./hybris/bin/custom/concerttours/resources/concerttours-spring.xml", "<context:component-scan base-package=\"concerttours\"/>"));
    assertEquals("Вы должны были добавить concerttours.service.impl.DefaultBandService.java", FileHelper.fileExistsAndContains(
        "./hybris/bin/custom/concerttours/src/concerttours/service/impl/DefaultBandService.java", "public class DefaultBandService implements BandService"));
    assertEquals("Вы должны были добавить concerttours.service.BandService.java", FileHelper.fileExistsAndContains(
        "./hybris/bin/custom/concerttours/src/concerttours/service/BandService.java", "публичный интерфейс BandService"));
}
  
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testServiceLayerClassesExist test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testServiceLayerClassesExist test
```

### Процедура

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Создать новый BandDAOИнтерфейс DAO подконцертные турырасширенияsrc/concerttours/daosпапка.

```

упаковка концертные туры.daos;
импорт java.util.List;
импорт концертные туры.модель.BandModel;

/**
 * Интерфейс для Band DAO, включающий различные операции для извлечения сохраненных объектов модели Band
 */
  
```

```
публичный интерфейс BandDAO
{
    /**
     * Возвращает список моделей полос, которые в настоящее время сохраняются. Если ни одна не найдена, возвращается пустой список.
     *
     * @return все группы в системе
     */
    Список<BandModel> findBands(); /**

     * Находит все полосы с указанным кодом. Если ничего не найдено, будет возвращен пустой список.
     *
     * @param код
     *         код для поиска групп
     * @return Все группы с указанным кодом.
     */
    Список<BandModel> findBandsByCode(код строки);
}
```

- Интерфейс состоит из методов, необходимых для теста DefaultBandDAOIntegrationTest.java.
- Комментарии описывают поведение как при успешном, так и при неудачном применении каждого метода.
- Вам не нуженохранятьМетод. Механизм экономии осуществляетсямодельСервис.
- findBandsByCodeвозвращает список, но поскольку вы установилиуникальный=истинавэлементы.xml, список будет содержать либо 1, либо 0 элементов.

Интерактивный ярлык SAP Commerce 123: если вы не можете создать новый интерфейс DAO самостоятельно или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/daos/BandDAO.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/daos/BandDAO.java

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/daos/BandDAO.java \$HYBRIS\_HOME\_DIR/hybris/bin/custom/c

эхо ж | xcopy %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\daos\BandDAO.java %HYBRIS\_HOME\_DIR%\hybris\b

. СоздайтеDefaultBandDAOреализация DAO в рамкахsrc/concerttours/daos/implнапаконцертные турырасширение.

```
упаковка concerttours.daos.impl; de.hybris.platform.servicelayer.search.FlexibleSearchQuery;
импорт de.hybris.platform.servicelayer.search.FlexibleSearchService; java.util.List;
импорт
импорт
импорт org.springframework.beans.factory.annotation.Autowired;
импорт org.springframework.stereotype.Component;
импорт концертные туры.daos.BandDAO;
импорт концертные туры.model.BandModel;

@Компонент(значение = "bandDAO")
открытый класс DefaultBandDAO реализует BandDAO {

    /**
     * Используйте SAP
     * Commerce FlexibleSearchService для выполнения запросов к базе данных
     */
    @Autowired
    частный ГибкийПоисковыйСервис гибкийПоисковыйСервис; /**

     * Находит все диапазоны, выполняя гибкий поиск с помощью {@link FlexibleSearchService}.
     */
    @Переопределить
    открытый список<BandModel> findBands() {

        // Построить запрос для гибкого поиска. final String
        queryString = //
            "ВЫБРАТЬ {p;" + BandModel.PK + "} //"
            + "ИЗ {" + BandModel._TYPECODE + " KAK p} ";
        окончательный запрос FlexibleSearchQuery = новый FlexibleSearchQuery(queryString);
        // Обратите внимание, что мы могли бы указать логику разбиения на страницы, указав переменные начала и количества (закомментировано ниже)
        // Это может обеспечить защиту от возврата очень больших объемов данных или перегрузки базы данных, когда, например, возвращаются миллионы
        элементов.
        // Поскольку мы знаем, что в этом варианте использования имеется только несколько постоянных полос, нам не нужно это
        предоставлять. //query.setStart(start);
        //запрос.setCount(count);
        // Возвращаем список BandModels.
        вернуть FlexibleSearchService.<BandModel> search(query).getResult();
    }
    /**
     * Находит все диапазоны по указанному коду, выполняя гибкий поиск с помощью {@link FlexibleSearchService}.
     */
    @Переопределить
    public List<BandModel> findBandsByCode(конечный код строки) {

        окончательная строка queryString = //
            "ВЫБРАТЬ {p;" + BandModel.PK + "} //"
            + "ИЗ {" + BandModel._TYPECODE + " KAK p} //"
            + "ГДЕ " + "(p;" + BandModel.CODE + ")=?код";
        окончательный запрос FlexibleSearchQuery = new FlexibleSearchQuery(queryString);
        query.addQueryParameter("код", код);
        вернуть FlexibleSearchService.<BandModel> search(query).getResult();
    }
}
```

- Для построения своих запросов DAO использует SAP Commerce Flexible Search Query. DefaultBandDAOкласс использует SAP Commerce Flexible Search Query для запросов. Если вы хотите сохранить новые элементы Band в DAO, используйтесохранятьМетод модельной службы. Однако, как показано на схеме в введении, вы можете вызвать модельную службусохранять вместо этого метод в классе обслуживания.
- Разделение логики DAO на отдельный POJO упрощает тестирование и делает разработку более понятной.
- НаверхDefaultBandDAOкласс, есть аннотация, предоставляемая фреймворком Spring: @Компонент (значение = "bandDAO").Эта аннотация сообщает Spring о необходимости определить компонент, экземпляр этого класса, называемыйгруппаDAOкоторые он может использовать для привязки зависимостей. Однако вам нужно указать Spring, что он должен сканировать этот пакет для поиска аннотированных bean-компонентов. Для этого добавьтеконтекст:компонент-сканированиемет в контексте приложения.

Интерактивный ярлык SAP Commerce 123: если вы не можете создать реализацию DAO самостоятельно или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/daos/impl/DefaultBandDAO.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/daos/impl/DefaultBandDAO.java

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/daos/impl/DefaultBandDAO.java \$HYBRIS\_HOME\_DIR/hybris/

```
echo f | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\daos\impl\DefaultBandDAO.java %HYBRIS_HOME_D
```

. Создайте BandService интерфейс обслуживания полосы под src\concerttours\service и пакаконцертные турырасширение.

```
упаковка концертные туры.сервис;
импорт java.util.List;
импорт концертные туры.модель.BandModel;

публичный интерфейс BandService {

    /**
     * Получает все диапазоны в системе.
     *
     * @return все группы в системе
     */
    Список<BandModel> getBands(); /**

     * Получает диапазон для указанного кода.
     *
     * @броски de.hybris.platform.servicelayer.exceptions.AmbiguousIdentifierException
     *           в случае, если для данного кода найдено более одной полосы
     * @броски de.hybris.platform.servicelayer.exceptions.UnknownIdentifierException
     *           в случае, если для данного кода не найдена полоса
     */
    BandModel getBandForCode(код строки);
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать интерфейс службы диапазона самостоятельно или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/service/BandService.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/service/BandService.java

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/service/BandService.java $HYBRIS_HOME_DIR/hybris/bin/cust
```

```
копировать /у %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\service\BandService.java %HYBRIS_HOME_DIR%\hybris\b
```

. Создайте DefaultBandService реализация службы полосы в соответствии csrsrc\concerttours\service\implпакаконцертные турырасширение.

```
упаковка concerttours.service.impl;
импорт de.hybris.platform.servicelayer.exceptions.AmbiguousIdentifierException;
импорт de.hybris.platform.servicelayer.exceptions.UnknownIdentifierException; java.util.List;
импорт
импорт org.springframework.beans.factory.annotation.Обязательно;
импорт concerttours.daos.BandDAO;
импорт концертные туры.модель.Группы;
импорт концертные туры.сервис.Сервис.Группы;

открытый класс DefaultBandService реализует BandService {

    частный BandDAO bandDAO; /**

     * Получает все диапазоны путем делегирования {@link BandDAO#findBands()}.
     */
    @Переопределить
    публичный список<BandModel> getBands() {

        вернуть bandDAO.findBands();
    }
    /**
     * Получает все полосы для указанного кода, делегируя {@link BandDAO#findBandsByCode(String)} и затем гарантируя
     * уникальность результата.
     */
    @Переопределить
    public BandModel getBandForCode(конечный строковый код) выдает AmbiguousIdentifierException, UnknownIdentifierException {

        окончательный список<BandModel> результат = bandDAO.findBandsByCode(код);
        если (результат.isEmpty())
        {
            throw new UnknownIdentifierException("Полоса с кодом '" + код + "' не найдена!");
        }
        иначе если (результат.размер() > 1)
        {
            throw new AmbiguousIdentifierException("Код полосы '" + код + "' не уникален, " + результат.size() + " полосы найдены!");
        }
        вернуть результат.получить(0);
    }
    @Необходимый
    public void setBandDAO (окончательный BandDAO bandDAO) {

        этот.bandDAO = bandDAO;
    }
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать реализацию службы диапазона самостоятельно или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/service/impl/DefaultBandService.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/service/impl/DefaultBandService.java

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/service/impl/DefaultBandService.java $HYBRIS_HOME_DIR/hy
```

```
копировать /у %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\service\impl\DefaultBandService.java %HYBRIS_HOME_D
```

. Настройте Spring для регистрации DAO и DefaultBandService как весенние бобы в <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttoursspring.xml

Заменить ранее использованный <фасоль xmlns=...>декларация с:

```
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:context="http://
       www.springframework.org/schema/context" xsi:schemaLocation="http://
       www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans-3.1.xsd http://
           www.springframework.org/schema/context
           http://www.springframework.org/schema/context/spring-context-3.1.xsd"
```

```
>
<контекст:компонент-сканирование базовый пакет="концертные туры"/>
```

Добавить:

```
<псевдоним = "defaultBandService" псевдоним = "bandService" />
<идентификатор компонента = "defaultBandService" класс = "concerttours.service.impl.DefaultBandService" > <имя
свойства = "bandDAO" ссылка = "bandDAO" />
</боб>
```

Интерактивный ярлык SAP Commerce 123: Если вы не можете зарегистрировать DAO и DefaultBandService самостоятельно или хотите пропустить этот шаг, замените `<HYBRIS_HOME_DIR>/hybris/bin/custom/concerttours/resources/concerttours-spring.xml` содержанием `<HYBRIS_HOME_DIR>/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-spring.xml`

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-spring.xml $HYBRIS_HOME_DIR/hybris/bin/custom/conce
```

копировать `/y %HYBRIS_HOME_DIR%\hybris123\src\main\WEB-INF\resources\concerttours\resources\concerttours-spring.xml %HYBRIS_HOME_DIR%\hybris\bin\custo`

. Перестройте SAP Commerce с помощью Ant.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ant очистить все
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и очистите все
```

. Запустите `testServiceLayerClassesExist` повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testServiceLayerClassesExist тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testServiceLayerClassesExist тест
```

. Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add . ; git commit -m "Расширить ServiceLayer"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Расширить ServiceLayer"
```

## Интеграционные тесты

Интеграционные тесты необходимы для демонстрации того, что ваша новая функциональность работает так, как и ожидалось. Они уведомляют вас, когда вы нарушаете существующее поведение, и, следовательно, могут помочь сократить количество ошибок.

Создайте интеграционный тест, который проверяет и демонстрирует некоторые из следующих ожидаемых вариантов поведения вашего DAO:

- Звонок `findBands` метод и что он должен возвращать, когда он находит и не находит данные
- Звонок `findBands` (код строки) метод и что он должен возвращать, когда находит данные и когда нет.
- Сохранение группы

Типичный интеграционный тест проверяет широкий спектр возможностей. Вы можете расширить тест, чтобы продемонстрировать и проверить, что метод успешно срабатывает, когда параметры находятся в пределах диапазона, а также что метод успешно и изящно терпит неудачу, когда параметры выходят за пределы диапазона. Ваши тестовые параметры могут включать любое или все из следующего:

- Неправильная комбинация заглавных и строчных букв
- Нулевое значение
- Пустая строка
- Пробел или недопустимые символы

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Уровень обслуживания](#)

Следующий:[Тести модулей](#)

Сопутствующая информация

[Интерфейс SAP Commerce Testweb](#)

## Проверьте свою услугу

Создайте интеграционный тест для вашего нового сервиса, затем запустите его и просмотрите результаты теста.

## Предпосылки

В ваш комплект документов включен следующий приемочный тест `Hybris123Tests.java`.

```
public void testServiceLayerIntegrationTest() выдает исключение {
    assertTrue( checkTestSuiteXMLMatches("(.*")testsuite ошибки="\"0\" неудачи="\"0\" (.*) имя=\"DefaultBandDAOIntegrationTest\" пакет=\"concerttours.da checkTestSuiteXMLMatches("(.*")testsuite ошибки="\"0\" неудачи="\"0\" (.*) имя=\"DefaultBandServiceIntegrationTest\" пакет=\"concerttours.service.im" )
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testServiceLayerIntegrationTest test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testServiceLayerIntegrationTest test
```

## Процедура

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Создайте DefaultBandDAOIntegrationTest.java Тест интеграции DAO в concerttours\tests\src\concerttours\daos\impl

```
упаковка concerttours.daos.impl; static org.junit.Assert.assertTrue;
импорт de.hybris.bootstrap.annotations.IntegrationTest;
импорт de.hybris.platform.core.Registry;
импорт de.hybris.platform.servicelayer.ServicelayerTransactionalTest;
импорт de.hybris.platform.servicelayer.model.ModelService; java.lang.InterruptedException;
импорт java.util.List;
импорт java.util.concurrent.TimeUnit;
импорт javax.annotation.Resource;
импорт org.junit.Assert;
импорт org.junit.After;
импорт org.junit.Do;
импорт org.junit.Test;
импорт org.springframework.jdbc.core.JdbcTemplate;
импорт концертные туры.модель.BandModel;

/**
 * Целью данного теста является демонстрация лучших практик и поведения DAO.
 *
 * Логика DAO вынесена в отдельный POJO. Пошаговое изучение этих схем покажет, как писать и выполнять
 * FlexibleSearchQueries — основа, на которой работает большинство гибридных DAO.
 */
@Интеграционный тест
открытый класс DefaultBandDAOIntegrationTest расширяет ServicelayerTransactionalTest {

    /** Поскольку это интеграционный тест, тестируемый класс (объект) внедряется сюда. */ @Resource
    частный BandDAO bandDAO;
    /** ModelService платформы используется для создания тестовых данных. */
    @Resource
    private ModelService modelService; /** Имя
    тестового диапазона.*/
    private static final String BAND_CODE = "ROCK-11"; /** Название
    тестовой группы.*/
    private static final String BAND_NAME = "Ladies of Rock"; /** История тестовой
    группы.*/
    private static final String BAND_HISTORY = "Полностью женская рок-группа, образованная в Мюнхене в конце 1990-х"; /** Продано
    альбомов*/
    private static final Long ALBUMS SOLD = Long.valueOf(1000L); @Before public void
    setUp() выдает исключение {
        пытаюсь {
            Thread.sleep(TimeUnit.SECONDS.toMillis(1));
            new JdbcTemplate(Registry.getCurrentTenant().getDataSource()).execute("CHECKPOINT");
            Thread.sleep(TimeUnit.SECONDS.toMillis(1));
        } catch (InterruptedException exc) {}
    }
    @Test
    public void bandDAOTest() {
        // проверяем, что наш тестовый диапазон еще не присутствует в базе данных
        List<BandModel> bandsByCode = bandDAO.findBandsByCode(BAND_CODE); assertTrue("No
        Band should be returns", bandsByCode.isEmpty()); // извлекаем все диапазоны, которые в
        данный момент находятся в базе данных
        Список<BandModel> allBands = bandDAO.findBands();
        окончательный размер int = allBands.size();
        // добавляем нашу тестовую группу в базу данных
        окончательная BandModel bandModel = modelService.create(BandModel.class);
        bandModel.setCode(BAND_CODE);
        bandModel.setName(BAND_NAME);
        bandModel.setHistory(BAND_HISTORY);
        bandModel.setAlbumSales(ALBUMS SOLD);
        modelService.save(bandModel);
        // проверяем, теперь мы получаем на одну полосу больше, чем раньше, и наша тестовая полоса находится в
        списке allBands = bandDAO.findBands();
        Assert.assertEquals(size + 1, allBands.size()); Assert.assertTrue("band not found",
        allBands.contains(bandModel)); // проверяем, можем ли мы найти нашу тестовую полосу
        по ее коду
        bandsByCode = bandDAO.findBandsByCode(BAND_CODE);
        Assert.assertEquals("Группа, которую мы только что сохранили, не найдена", 1, bandsByCode.size()); Assert.assertEquals("Неверный атрибут кода полученной
        группы", BAND_CODE, bandsByCode.get(0).getCode()); Assert.assertEquals("Неверный атрибут имени полученной группы", BAND_NAME,
        bandsByCode.get(0).getName()); Assert.assertEquals("Неверный атрибут albumSales полученной группы", ALBUMS SOLD, bandsByCode.get(0).getAlbumSales());
        Assert.assertEquals("Неверный атрибут истории полученной группы", BAND_HISTORY, bandsByCode.get(0).getHistory());
    }
    @Test
    public void testFindBands_EmptyStringParam() {
        // вызов findBandsByCode() с пустой строкой - не возвращает никаких результатов final
        List<BandModel> bands = bandDAO.findBandsByCode(""); Assert.assertTrue("Не следует
        возвращать ни одну полосу", bands.isEmpty());
    }
    @Test(ожидаемый = IllegalArgumentException.class) public void
    testFindBands_NullParam()
    {
        // вызов findBandsByCode с null должен выдать исключение IllegalArgumentException
        bandDAO.findBandsByCode(null); //возвращаемое значение метода не зафиксировано
    }
}
```

```
@После public void tearDown() {
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать тестовый класс самостоятельно или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/testsrc/concerttours/daos/impl/DefaultBandDAOIntegrationTest.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/daos/impl/DefaultBandDAOIntegrationTest.java

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/daos/impl/DefaultBandDAOIntegrationTest.java \$HYBR

```
echo f | xcopy %HYBRIS_HOME_DIR%hybris123\src\main\webapp\resources\concerttours\testsrc\concerttours\daos\impl\DefaultBandDAOIntegrationTest.
```

. Создайте второй интеграционный тест для проверки BandService, создав тестовый класс с именем concerttours.service.impl.DefaultBandServiceIntegrationTest под тестируемым классом DefaultBandService.

```
упаковка concerttours.service.impl;
импорт static org.junit.Assert.assertEquals;
импорт static org.junit.Assert.assertNotNull;
импорт de.hybris.bootstrap.annotations.IntegrationTest;
импорт de.hybris.platform.core.model.product.ProductModel;
импорт de.hybris.platform.servicelayer.ServicelayerTest;
импорт de.hybris.platform.servicelayer.exceptions.UnknownIdentifierException;
импорт de.hybris.platform.servicelayer.model.ModelService;
импорт de.hybris.platform.variants.model.VariantProductModel;
импорт java.lang.InterruptedException;
импорт java.util.List;
импорт java.util.Set;
импорт javax.annotation.Resource;
импорт org.junit.Assert;
импорт org.junit.Do;
импорт org.junit.Test;
импорт concerttours.model.BandModel;
импорт concerttours.service.BandService;
импорт java.util.concurrent.TimeUnit;
импорт de.hybris.platform.core.Registry;
импорт org.springframework.jdbc.core.JdbcTemplate;
```

@Интеграционный тест  
открытый класс DefaultBandServiceIntegrationTest расширяет ServicelayerTest {

```
@Ресурс
частная BandService bandService;
@Ресурс
частная модельСервис модельСервис;
/** Тестовый диапазон */ private
BandModel bandModel; /** Имя
тестового диапазона.*/
private static final String BAND_CODE = "101-JAZ"; /** Название
тестовой группы.*/
private static final String BAND_NAME = "Tight Notes"; /** История
тестовой группы.*/
private static final String BAND_HISTORY = "Новый современный джазовый коллектив из 7 человек из Лондона, сформированный в 2015 году"; /** Альбомы,
проданные тестовой группой.*/
частный статический финальный Long ALBUMS_SOLD = Long.valueOf(10L);
@Before
публичный недействительный setUp()
{
    пытаться {
        Thread.sleep(TimeUnit.SECONDS.toMillis(1));
        new JdbcTemplate(Registry.getCurrentTenant().getDataSource()).execute("CHECKPOINT");
        Thread.sleep(TimeUnit.SECONDS.toMillis(1));
    } catch (InterruptedException exc) {}
    // Этот экземпляр BandModel будет использоваться тестами bandModel =
    modelService.create(BandModel.class); bandModel.setCode(BAND_CODE);

    bandModel.setName(ИМЯ_ГРУППЫ);
    bandModel.setAlbumSales(ПРОДАННЫЕ_АЛЬБОМЫ);
    bandModel.setHistory(ХРОНИКА_ГРУППЫ);
}
@Test(ожидаемый = UnknownIdentifierException.class) public void
testFailBehavior()
{
    bandService.getBandForCode(BAND_CODE);
}
/**
 * Этот тест проверяет и демонстрирует, что метод getAllBand сервиса вызывает метод getAllBand DAO и
 * возвращает полученные от него данные.
 */
@Test
общественный недействительный testBandService()
{
    Список<BandModel> bandModels = bandService.getBands(); конечный
размер int = bandModels.size();
modelService.save(bandModel); bandModels =
bandService.getBands(); assertEquals(size + 1,
bandModels.size());
assertEquals("Обнаружена неожиданная полоса", bandModel, bandModels.get(bandModels.size() - 1)); final BandModel
persistedBandModel = bandService.getBandForCode(BAND_CODE); assertNotNull("Полоса не найдена",
persistedBandModel);
assertEquals("Найдена другая полоса", bandModel, persistedBandModel);
}
/**
 * Этот тест проверяет и демонстрирует, что метод getAllBand сервиса вызывает метод getAllBand DAO и
 * возвращает полученные от него данные.
 */
@Test
public void testBandServiceTours() выдает исключение {

    createCoreData();
importCsv("/impex/concerttours-bands.impex", "utf-8"); importCsv("/impex/
concerttours-yBandTour.impex", "utf-8"); final BandModel band =
bandService.getBandForCode("A001"); assertNotNull("Группа не найдена",
band);
final Set<ProductModel> tours = band.getTours();
assertNotNull("Тур не найден", tours);
}
```

12/3/24, 11:01 утра

```
Assert.assertEquals("не найдено ни одного тура", 1, tours.size()); final Object[] objects = new Object[5]; final Collection<VariantProductModel> concerts = ((ProductModel) tours.toArray(objects)[0]).getVariants(); assertNotNull("Тип не найден", tours);
```

```
        Assert.assertEquals("не найдено ни одного тура", 6, concerts.size());  
    }
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать тестовый класс самостоятельно или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/testsrc/concerttours/service/impl/DefaultBandServiceIntegrationTest.javac содержанием

<HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/service/impl/DefaultBandServiceIntegrationTest.

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/service/impl/DefaultBandServiceIntegrationTest.java

```
echo f | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\testsrc\concerttours\service\impl\DefaultBandServiceIntegrationTest
```

.Перестройте SAP Commerce с помощью Ant.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ant очистить все
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и очистите все
```

.Инициализируйте своего тестового арендатора.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; муравей юнитинит
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и ant yunitinit
```

.Запустите интеграционные тесты и убедитесь, что оба новых интеграционных теста пройдены.

```
муравей интеграционные тесты - Dtestclasses.packages="concerttours.*"
```

```
муравей интеграционные тесты - Dtestclasses.packages=концертные туры.*
```

.Посмотреть результаты теста можно на <HYBRIS\_HOME\_DIR>/hybris/log/junit/index.html

.Запустите testServiceLayerIntegrationTest повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testServiceLayerIntegrationTest test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testServiceLayerIntegrationTest test
```

.Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add . ; git commit -m "Протестируйте свою службу"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Протестируйте свою службу"
```

## Тесты модулей

Вы можете моделировать зависимости для выполнения модульных тестов, которые выполняются независимо от платформы SAP Commerce.

Интеграционный тест показывает, что вы успешно достигли ожидаемого поведения классов, реализующих BandServiceинтерфейс. Однако это не настояще тестирование ожидаемой функциональности вашего конкретного DefaultBandServiceкласса. Для этого вам нужно смоделировать зависимости, BandDAOв этом случае, используя Mockito в новом классе модульного теста, называемом DefaultBandServiceUnitTest. Поскольку это имитирует реальный DAO, тест не требует доступа к слою сохранения SAP Commerce и, следовательно, не требует расширения ServiceLayerТранзакционный Тест. Вместо этого это простой POJO, который будет работать очень быстро.

Вы проверите, что ожидаемые вызовы выполняются из BandServiceк BandDAOИнтерфейс. Нет реализации BandDAOнеобходимо при моделировании BandDAOИнтерфейс.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Интеграционные тесты](#)

Следующий:[Фасадный слой](#)

Сопутствующая информация

[Spring Framework в SAP Commerce](#)

[Работа с ServiceLayer](#)

## Протестируйте новую услугу с помощью модульного теста

Используйте Mockito для создания симуляции BandDAOреализация, которую обычно использует служба.

## Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.java

```
public void testServiceLayerUnitTest() {  
    assertEquals( checkTestSuiteXMLMatches("(.*<testsuite ошибки=""0"" неудачи=""0"" (.*) имя=""DefaultBandServiceUnitTest"" пакет=""concerttours.serv
```

}

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testServiceLayerUnitTest тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testServiceLayerUnitTest тест
```

## Процедура

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Создайте новый DefaultBandServiceUnitTest.java класс модульного теста в `tests\src\java\com\hybris\concerttours\service\impl`

```
/*
 * [y] hybris Платформа
 *
 * Авторские права (c) 2000-2017 SAP SE
 * Все права защищены.
 *
 * Данное программное обеспечение является конфиденциальной и частной информацией SAP.
 * Hybris («конфиденциальная информация»). Вы не должны разглашать такую информацию.
 * Конфиденциальная информация и использовать ее только в соответствии с
 * условиями лицензионного соглашения, заключенного вами с SAP Hybris.
 */
упаковка concerttours.service.impl;
импорт static comunit.Assert.assertEquals;
импорт static org.mockito.Mockito.mock;
импорт static org.mockito.Mockito.when;
импорт de.hybris.bootstrap.annotations.UnitTest; java.util.Arrays;
импорт
импорт java.util.Collections;
импорт java.util.List;
импорт org.junit.Do;
импорт org.junit.Test;
импорт конкретные туры.daos.BandDAO;
импорт конкретные туры.model.BandModel;

/**
 * Этот тестовый файл тестирует и демонстрирует поведение методов BandService getAllBand, getBand и saveBand.
 *
 * У нас уже есть отдельный файл для тестирования Band DAO, и мы не хотим, чтобы этот тест неявно проверял это
 * дополнение к BandService. Таким образом, этот тест имитирует Band DAO, оставляя нам возможность тестируть Service в изоляции,
 * поведение которого должно быть просто для обравчивания вызовов к DAO: переадресация вызовов к нему и передача результатов ему
 * получает из этого.
 */
@UnitTest
публичный класс DefaultBandServiceUnitTest
{
    частный DefaultBandService bandService; BandDAO
    частный bandDAO;
    частный BandModel bandModel;
    /** Название тестового диапазона. */
    private static final String BAND_CODE = "Ch00X"; /** Имя
    тестового диапазона.*/
    private static final String BAND_NAME = "Singers All"; /** История тестовой
    группы.*/
    private static final String BAND_HISTORY = "Средневековый хор, образованный в 2001 году, базирующийся в Мюнхене, известный аутентичными монастырскими песнопениями"; @Before
    public void setUp()
    {
        // Мы будем тестируировать BandServiceImpl — реализацию BandService bandService = new
        DefaultBandService();
        // Чтобы не тестируировать DAO неявно, мы создадим макет DAO с помощью Mockito bandDAO =
        mock(BandDAO.class);
        // и внедрить этот фиктивный DAO в BandService
        bandService.setBandDAO(bandDAO);
        // Этот экземпляр BandModel будет использоваться в тестах bandModel =
        new BandModel();
        bandModel.setCode(КОД_ГРУППЫ);
        bandModel.setName(НАЗВАНИЕ_ГРУППЫ);
        bandModel.setAlbumSales(1000L);
        bandModel.setHistory(ИСТОРИЯ_ГРУППЫ);
    }
    /**
     * Этот тест проверяет и демонстрирует, что метод getAllBands сервиса вызывает метод getBands DAO и возвращает
     * данные, которые он получает от него.
     */
    @Test
    общественный недействительный testGetAllBands () {
        // Мы создаем данные, которые должен возвращать имитированный DAO при вызове final
        List<BandModel> bandModels = Arrays.asList(bandModel);
        //Используем Mockito и сравниваем результаты
        when(bandDAO.findBands()).thenReturn(bandModels);
        // Теперь мы вызываем метод BandService.getBands(), который, как мы ожидаем, вызовет метод DAO findBands() final List<BandModel>
        result = bandService.getBands();
        // Затем мы проверяем, что результаты, возвращаемые службой, соответствуют результатам, возвращаемым имитированным DAO
        assertEquals("We should find one", 1, result.size());
        assertEquals("И должно быть равно тому, что вернула имитация", bandModel, result.get(0));
    }
    @Test
    общественный void testGetBand() {
        // Сообщаем Mockito, что мы ожидаем вызова getBand() DAO, и, если это произойдет, Mockito должен вернуть экземпляр BandModel
        when(bandDAO.findBandsByCode(BAND_CODE)).thenReturn(Collections.singletonList(bandModel));
        // Мы вызываем getBandForCode() сервиса, который, как мы ожидаем, вызовет findBandsByCode() DAO final BandModel result =
        bandService.getBandForCode(BAND_CODE);
        // Затем мы проверяем, что результат, возвращенный службой, совпадает с результатом, возвращенным DAO
        assertEquals("И должно быть равно тому, что вернула имитация", bandModel, result.get(0));
    }
}
```

12/3/24, 11:01 утра

```
    assertEquals("Band должен быть равен() тому, что вернула имитация", bandModel, result);
}
```

Интерактивный ярлык SAP Commerce 123: если вам не удается создать DefaultBandServiceUnitTest.java класс модульного теста или вы хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/testsrc/concerttours/service/impl/DefaultBandServiceUnitTest.java содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/service/impl/DefaultBandServiceUnitTest.java

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/service/impl/DefaultBandServiceUnitTest.java $HYBRIS_
```

```
копировать %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\testsrc\concerttours\service\impl\DefaultBandServiceUnitTest.java %HYBRIS
```

. Перестройте SAP Commerce с помощью Ant.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ant очистить все
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и очистите все
```

. Запустите модульный тест.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ant unittests -Dtestclasses.packages="concerttours.*"
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и ant unittests -Dtestclasses.packages=concerttours.*
```

. Посмотреть результаты теста можно на <HYBRIS\_HOME\_DIR>/hybris/log/junit/index.html

. Запустите testServiceLayerUnitTest повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testServiceLayerUnitTest тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testServiceLayerUnitTest тест
```

. Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add .; git commit -m "Протестируйте свой новый сервис с помощью модульного теста"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Протестируйте свою новую службу с помощью модульного теста"
```

## Фасадный слой

Фасад — это уровень абстракции, который предоставляет упрощенный интерфейс для базовой реализации.

В предыдущих разделах вы видели, что SAP Commerce генерирует классы моделей Java, которые представляют различные типы элементов, хранящихся в базе данных. Эти классы моделей — это то, что вы передаете в качестве аргументов различным службам Java на уровне служб SAP Commerce. Тем не менее, бывают случаи, когда классы моделей становятся громоздкими:

- Когда вам нужен более простой или удобный формат для отображения некоторых данных в JSP
- Когда вам нужен сериализуемый набор объектов для отправки в другую систему
- Когда вы хотите запретить клиентскому коду изменять атрибуты в объекте класса модели напрямую

В этих случаях вам необходимо более простое представление данных в классах модели. Это представление является целью объекта передачи данных. Кроме того, если есть общая последовательность вызовов методов, которые клиент должен выполнить в отношении объекта службы, имеет смысл объединить последовательность в один вызов. Вы делаете эти упрощенные вызовы с помощью объекта фасада.

Классы фасада упрощают вызовы, сделанные для ваших классов сервисов. Они используют более простые объекты Java (POJO) в качестве объектов аргументов и результатов вместо классов моделей SAP Commerce. На этом этапе вы создаете новый BandFacadeDescriptor.

Родительская тема: [Изучите тип SAP Commerce](#)

Предыдущий: [Тесты модулей](#)

Следующий: [Передняя часть](#)

## Расширить фасадный слой

Создавайте фасадные классы, которые предоставляют упрощенный интерфейс для вашей логики расширения.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест Hybris123Tests.java:

```
public void testFacadeLayerOk() {
    assertTrue(checkTestSuiteXMLMatches(".*тестсюит ошибки=\"0\" неудачи=\"0\" (.*) имя=\"DefaultBandFacadeIntegrationTest\" пакет=\"concerttours\"")
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testFacadeLayerOk тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testFacadeLayerOk тест
```

## Процедура

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Объявляем объекты передачи данных в concerttours-beans.xml.

Добавьте следующее к concerttours-beans.xml в папке ресурсов расширения concerttours.

```
<класс компонента = "concerttours.data.TourSummaryData">
    <description>Объект данных для краткого описания тура, не имеющий эквивалента в системе типов</description> <property name =
    "id" тип = "String" />
    <имя свойства = "tourName" тип = "Строка" /> <имя свойства =
    "numberOfConcerts" тип = "Строка" />
</боб>
<класс компонента = "concerttours.data.BandData" >
    <description>Объект данных, представляющий Band</description> <property
    name = "id" type = "String" />
    <имя свойства = "name" тип = "String" /> <имя свойства =
    "description" тип = "String" /> <имя свойства = "albumsSold" тип =
    "Long" />
    <имя свойства = "genres" тип="java.util.List<Строка>" />
    <имя свойства = "tours" тип="java.util.List<concerttours.data.TourSummaryData>" />
</боб>
<класс компонента = "concerttours.data.ConcertSummaryData">
    <description>Объект данных для краткого содержания концерта</description>
    <property name = "id" type = "String" />
    <имя свойства = "data" тип = "java.util.Date" /> <имя свойства =
    "место проведения" тип = "Строка" /> <имя свойства = "тип" тип =
    "Строка" />
</боб>
<класс компонента = "concerttours.data.TourData" >
    <description>Объект данных, представляющий тип</description> <property
    name = "id" type = "String" />
    <имя свойства = "tourName" тип = "Строка" /> <имя свойства =
    "description" тип = "Строка" />
    <имя свойства = "concerts" тип="java.util.List<concerttours.data.ConcertSummaryData>" /> </bean>
```

Вы определяете бины и перечисления в XML-файле, который генератор кода принимает в качестве входных данных. Главное преимущество этого метода заключается в том, что вы можете объединить атрибуты нескольких расширений в один и тот же DTO таким же образом, как определения типов элементов объединяются из нескольких элементов.xml. Это делает ваш фасадный слой более расширяемым.

### -Примечание

- Вы создаете сводку DTO для туров, которая просто включает идентификатор, название и количество концертов. Вы можете использовать это на страницах, где вам нужно перечислить туры или иным образом отобразить краткие сведения о туре.
- В BandDatabeans, вы используете обобщенные типы Java для ограничения типов объектов, возвращаемых в списках связанных жанров и туров.
- В данные турабы, вы ссылаетесь на ConcertSummaryDataобы, которые вы определили непосредственно перед этим в le.

Интерактивный ярлык SAP Commerce 123: если вам не удается создать concerttours-beans.xml в себя или вы хотите пропустить этот шаг, замените

```
<HYBRIS_HOME_DIR>/hybris/bin/custom/concerttours/resources/concerttours-beans.xml с содержанием <HYBRIS_HOME_DIR>/hybris123/src/main/webapp/
resources/concerttours/resources/concerttours-beansWithFacadeLayer.xml.
```

```
то же самое $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-beansWithFacadeLayer.xml $HYBRIS_HOME_DIR/hybris
```

```
копировать /у %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\resources\concerttours-beansWithFacadeLayer.xml %HYBRIS_HOME_DIR%\hy
```

. Создайте новые DTO из этого определения XML, вызывав муравей очистить все из <HYBRIS\_HOME\_DIR>/hybris/bin/платформа каталог.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ant очистить все
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и очистите все
```

Команда ant генерирует модель и классы DTO в платформа/bootstrap/gensrc каталог. После завершения взгляните на недавно созданные классы объектов передачи данных: BandData, TourData, ConcertSummaryData и другие.

. Создайте интерфейсы фасада Band и Tour в <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/facades/.

```
упаковка концертные туры.фасады;
импорт java.util.List;
импорт concerttours.data.BandData;

публичный интерфейс BandFacade {
    BandData getBand (имя строки);
    List<BandData> getBands();
}

упаковка концертные туры.фасады;
импорт concerttours.data.TourData;

публичный интерфейс TourFacade {
    TourData getTourDetails(конечная строка tourId);
}
```

Как и сервисы SAP Commerce, фасады создаются как Spring-бины, реализующие определенный интерфейс Java. Цель этих фасадов — предоставить API бизнес-уровня для использования вызывающими клиентами. В этом случае они извлекают сведения о Bands и Tours.

Интерактивный ярлык SAP Commerce 123: если вы не можете создать интерфейсы фасада самостоятельно или хотите пропустить этот шаг, скопируйте <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/facades/\*Facade.java в <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/facades/.

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/facades/TourFacade.java \$HYBRIS\_HOME\_DIR/hybris/bin/cu

эхо < | xcopy %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\facades\Facade.java %HYBRIS\_HOME\_DIR%\hybris\bin\custom\concerttours\src\concerttours\facades\impl\.

. Создайте реализации фасада Band и Tour в <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/facades/impl/.

```
упаковка concerttours.facades.impl;
импорт de.hybris.platform.core.model.product.ProductModel;
импорт java.util.ArrayList;
импорт java.util.List;
импорт org.springframework.beans.factory.annotation.Required;
импорт
импорт concerttours.data.TourSummaryData;
импорт concerttours.enums.MusicType;
импорт concerttours.facades.BandFacade;
импорт concerttours.model.BandModel;
импорт concerttours.service.BandService;
импорт java.util.Locale;
```

открытый класс DefaultBandFacade реализует BandFacade {

```
частный BandService bandService;
@Override
публичный список<BandData> getBands() {

    окончательный список<BandModel> bandModels = bandService.getBands();
    окончательный список<BandData> bandFacadeData = new ArrayList<>(); для
    (окончательный BandModel sm : bandModels)
    {
        окончательный BandData sfd = новый
        BandData(); sfd.setId(sm.getCode());
        sfd.setName(sm.getName());
        sfd.setDescription(sm.getHistory());
        sfd.setAlbumsSold(sm.getAlbumSales());
        bandFacadeData.add(sfd);
    }
    вернуть bandFacadeData;
}
```

```
@Переопределить
public BandData getBand(конечное имя строки) {
```

```
если (имя == null) {
    throw new IllegalArgumentException("Название группы не может быть пустым");
}
окончательная BandModel band = bandService.getBandForCode(name); если
(band == null)
{
    вернуть ноль;
}

// Создать список жанров
окончательный список<String> жанры = новый ArrayList<>;
если (band.getTypes() != null)
{
    для (финальный ТипМузыки musicType : band.getTypes()) {
        жанры.добавить(musicType.получитьКод());
    }
}
// Создаем список TourSummaryData из совпадений final
List<TourSummaryData> tourHistory = new ArrayList<>(); if (band.getTours() != null)
```

```
{
    для (финальный тип ProductModel: band.getTours()) {

        окончательная сводка TourSummaryData = new TourSummaryData();
        summary.setId(tour.getCode());
        summary.setTourName(tour.getName(Locale.ENGLISH));
        // делаем большое предположение, что все варианты являются концертами и игнорируем каталоги продуктов
        summary.setNumberOfConcerts(Integer.toString(tour.getVariants().size())); tourHistory.add(summary);
    }
}
```

```
// Теперь мы можем создать объект передачи BandData Final
BandData BandData = new BandData();
BandData.setId(band.getCode());
BandData.setName(band.getName());
bandData.setAlbumsSold(band.getAlbumSales());
bandData.setDescription(band.getHistory());
BandData.setGenres(жанры);
BandData.setTours(История тура);
вернуть данные полосы;
```

```
}
```

```
@Необходимый
public void setBandService(final BandService bandService) {
```

```
этот.bandService = bandService;
```

```
}
```

```
упаковка concerttours.facades.impl;
импорт de.hybris.platform.core.model.product.ProductModel;
импорт de.hybris.platform.product.ProductService;
импорт de.hybris.platform.variants.model.VariantProductModel; java.util.ArrayList;
импорт
импорт java.util.List;
импорт org.springframework.beans.factory.annotation.Required;
импорт concerttours.data.ConcertSummaryData;
импорт concerttours.data.Данные тура;
импорт concerttours.enums.Тип концерта;
импорт concerttours.Фасад тура;
импорт concerttours.модель.Модель концерта;
```

открытый класс DefaultTourFacade реализует TourFacade {

```
частный ProductService productService; @Override
```

```
публичные данные типа getTourDetails(конечная строка tourId)
```

```

{
    если (tourId == null) {
        throw new IllegalArgumentException("Идентификатор тура не может быть пустым");
    }
    конечный ProductModel продукт = productService.getProductForCode(tourId); если (продукт ==
    null)
    {
        вернуть ноль;
    }
    // Создаем список ConcertSummaryData из совпадений final
    List<ConcertSummaryData> concerts = new ArrayList<>(); if (product.getVariants() !=
    null)
    {
        для (окончательный вариант VariantProductModel : product.getVariants()) {
            если (вариант instanceof ConcertModel) {

                финал ConcertModel концерт = (ConcertModel) варианта; финал
                ConcertSummaryData резюме = new ConcertSummaryData();
                summary.setId(concert.getCode());
                summary.установитьДату(концерт.получитьДату()); summary.установитьМесто
                проведения(концерт.получитьМестоПроведения());
                summary.setTipo(concert.getConcertType() == ConcertType.OPENAIR ? "На открытом воздухе" : "В помещении");
                concerts.add(summary);
            }
        }
    }
    // Теперь мы можем создать объект передачи TourData final
    TourData tourData = new TourData();
    tourData.setId(product.getCode());
    tourData.setTourName(product.getName());
    tourData.setDescription(product.getDescription());
    tourData.setConcerts(концерты);
    вернуть данные тура;
}
@Необходимый
public void setProductService(final ProductService productService) {
    этот.productService = productService;
}
}

```

Реализации фасада выполняют вызовы к службам в ServiceLayer, в частности BandService и ProduktService для обеспечения необходимой функциональности.

#### -Примечание

Обратите внимание, что вы жестко кодируете заполнение объектов передачи данных из объектов модели. Для простых сценариев этот подход совершенно хороший. Тем не менее, для более сложных сценариев платформа SAP Commerce имеет понятие преобразователей. Преобразователи — это объекты, которые делегируют эту задачу цепочке объектов-заполнителей. Для получения более подробной информации см. [Конвертеры и пополнители](#).

Интерактивный ярлык SAP Commerce 123: если вы не можете создать реализации фасада самостоятельно или хотите пропустить этот шаг, скопируйте <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/facades/impl/DefaultBandFacade.java и <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/facades/impl/DefaultTourFacade.java в <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/facades/impl/каталог.

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/facades/impl/DefaultBandFacade.java \$HYBRIS\_HOME\_DIR/h

echo f | xcopy %HYBRIS\_HOME\_DIR%hybris123src\main\webapp\resources\concerttours\src\concerttours\facades\impl\\*Facade.java %HYBRIS\_HOME\_DIR%

.Объявляем фасады в концертные туры-весна.xml так, чтобы они были подключены к нашему расширению.

```

<имя псевдонима = "defaultBandFacade" псевдоним = "bandFacade" />
<bean id = "defaultBandFacade" class = "concerttours.facades.impl.DefaultBandFacade" >
    <имя свойства = "bandService" ref = "bandService" /> </bean>

```

```

<псевдоним = "defaultTourFacade" псевдоним = "tourFacade" />
<bean id = "defaultTourFacade" class = "concerttours.facades.impl.DefaultTourFacade" >
    <имя свойства = "productService" ref = "productService" /> </bean>

```

Интерактивный ярлык SAP Commerce 123: если вы хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttoursspring.xml на содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-springwithFacade.xml.

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-spring-withFacade.xml \$HYBRIS\_HOME\_DIR/hybris/bi

копировать /у %HYBRIS\_HOME\_DIR%hybris123src\main\webapp\resources\concerttours\resources\concerttours-spring-withFacade.xml %HYBRIS\_HOME\_DIR%hybris

.Создайте тест интеграции фасадов в <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/testsrc/concerttours/facades/impl/.

Хорошей практикой является написание теста, который может проверить правильность поведения ваших реализаций фасада.

```

упаковка концертные туры.фасады.импл;
импорт static org.junit.Assert.assertEquals;
импорт static org.junit.Assert.assertNotNull;
импорт de.hybris.bootstrap.annotations.IntegrationTest;
импорт de.hybris.platform.servicelayer.ServicelayerTransactionalTest;
импорт de.hybris.platform.servicelayer.exceptions.UnknownIdentifierException;
импорт de.hybris.platform.servicelayer.model.ModelService;
импорт java.lang.InterruptedException;
импорт java.util.List;
импорт java.util.concurrent.TimeUnit;
импорт de.hybris.platform.core.Registry;
импорт org.springframework.jdbc.core.JdbcTemplate;
импорт javax.annotation.Resource;
импорт org.junit.After;
импорт org.junit.Before;
импорт org.junit.Test;
импорт concerttours.data.BandData;
импорт

```

```

импорт концертных туров.модель.BandModel;

/**
 * Этот тестовый файл тестирует и демонстрирует поведение методов BandFacade getAllBands и getBand.
 */
@Интеграционный тест
открытый класс DefaultBandFacadeIntegrationTest расширяет ServicelayerTransactionalTest {

    @Ресурс
    частный BandFacade bandFacade;
    @Resource
    частный ModelService modelService; /**
    Тестовая полоса */
    private BandModel bandModel; /**
    Имя тестового диапазона */
    private static final String BAND_CODE = "101-JAZ"; /** Название
    тестовой группы */
    private static final String BAND_NAME = "Tight Notes"; /** История
    тестовой группы */
    private static final String BAND_HISTORY = "Новый современный джазовый коллектив из 7 человек из Лондона, сформированный в 2015 году"; /** Альбомы,
    проданные тестовой группой */
    частный статический финальный Long ALBUMS SOLD = Long.valueOf(10L);
    @Before
    публичный недействительный setUp() {
        пытаться {
            Thread.sleep(TimeUnit.SECONDS.toMillis(1));
            new JdbcTemplate(Registry.getCurrentTenant().getDataSource()).execute("CHECKPOINT");
            Thread.sleep(TimeUnit.SECONDS.toMillis(1));
        } catch (исключение InterruptedException exc) {}
        // Этот экземпляр BandModel будет использоваться тестами bandModel =
        modelService.create(BandModel.class); bandModel.setCode(BAND_CODE);

        bandModel.setName(ИМЯ_ДИАПАЗОНА);

        bandModel.setHistory(BAND_HISTORY);
        bandModel.setAlbumSales(ALBUMS SOLD);
    }
    /**
     * Тестирует поведение исключения, получая полосу, которая не существует
     */
    @Test(ожидаемый = UnknownIdentifierException.class) public void testInvalidParameter() {
        bandFacade.getBand(ИМЯ_BAND);
    }
    /**
     * Тестирует поведение исключения, передавая нулевой параметр
     */
    @Test(ожидаемый = IllegalArgumentException.class) public void testNullParameter() {
        bandFacade.getBand(null);
    }
    /**
     * Тестирует и демонстрирует методы Facade
     */
    @Test
    public void testBandFacade() {
        List<BandData> bandListData = bandFacade.getBands(); AssertNotNull
        (bandListData);
        окончательный размер int = bandListData.size();
        modelService.save(bandModel);
        BandListData = BandFacade.getBands();
        AssertNotNull(bandListData);
        assertEquals(размер + 1, bandListData.size());
        assertEquals(КОД_ДИАПАЗОНА, bandListData.get(size).getId());
        assertEquals(ИМЯ_ГРУППЫ, bandListData.get(size).getName());
        assertEquals(ПРОДАННЫЕ_АЛЬБОМЫ, bandListData.get(size).getAlbumsSold());
        assertEquals(ИСТОРИЯ_ГРУППЫ, bandListData.get(size).getDescription());
        окончательные BandData persistedBandData = bandFacade.getBand(BAND_CODE);

        assertEquals(КОД_ГРУППЫ, persistedBandData.getId());
        assertEquals(НАЗВАНИЕ_ГРУППЫ, persistedBandData.getName());
        assertEquals(ПРОДАННЫЕ_АЛЬБОМЫ, persistedBandData.getAlbumsSold());
        assertEquals(ИСТОРИЯ_ГРУППЫ, persistedBandData.getDescription());
    }
    @После public void teardown() {
    }
}

```

## -Примечание

- Добавьте к классу аннотацию @Интеграционный тест так что его можно будет обнаружить с помощью встроенных целей тестирования Ant и расширить ServicelayerТранзакционныйТест чтобы среда SAP Commerce была запущена и доступна во время выполнения теста.
- Используйте Модель Сервис для создания новых экземпляров классов моделей, чтобы новые объекты уже были прикреплены к контексту сохранения. Если бы вы использовали стандартный оператор Java new, вам пришлось бы явно прикрепить объект к контексту сохранения перед его сохранением.

Интерактивный ярлык SAP Commerce 123: если вы не можете создать интеграционный тест самостоятельно или хотите пропустить этот шаг, скопируйте

<HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/facades/impl/DefaultBandFacadeIntegrationTest.j к <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/testsrc/concerttours/facades/impl/каталог.

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/facades/impl/DefaultBandFacadeIntegrationTest.java

echo f | xcopy %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\testsrc\concerttours\facades\impl\DefaultBandFacadeIntegrationTest.j

. Создайте модульный тест в <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/testsrc/concerttours/facades/impl/ для тестирования DefaultBandFacade в изоляции.

Назовите тестовый класс DefaultBandFacadeUnitTest, и сохраните его в концертные туры.фасады.импл.пакет под тесты гспапаконцертные туры расширение.

пакет concerttours.facades.impl;

импорт staticский org.mockito.Mockito.mock;  
импорт staticкий org.mockito.Mockito.когда;

12/3/24, 11:01 утра

```
импорт de.hybris.bootstrap.annotations.UnitTest;
импорт de.hybris.platform.servicelayer.model.ModelService; java.util.ArrayList;
импорт
импорт java.util.List;
импорт java.util.Locale;
импорт org.junit.Утверждение;
импорт org.junit.До;
импорт org.junit.Test;
импорт concerttours.data.BandData;
импорт concerttours.model.BandModel;
импорт concerttours.service.BandService;

@UnitTest
публичный класс DefaultBandFacadeUnitTest
{
    частный DefaultBandFacade bandFacade;
    частный ModelService modelService;
    частный BandService bandService;
    частная статическая финальная строка BAND_CODE = "ROCK-11"; частная
    статическая финальная строка BAND_NAME = "Ladies of Rock"; частная
    статическая финальная Long ALBUMS SOLD = Long.valueOf(42000);
private static final String BAND_HISTORY = "Полностью женская рок-группа, образованная в Мюнхене в конце 1990-х"; // Удобный метод
для возврата списка групп
частный список<BandModel> dummyDataBandList() {

    окончательный список<BandModel> полосы = new ArrayList<BandModel>();
    окончательный список BandModel полосы = configTestBand();
    полосы.добавить(полоса);
    возвратные полосы;
}
// Удобный метод для возврата настроенного тестового диапазона private
BandModel configTestBand()
{
    окончательная BandModel band = new BandModel();
    band.setCode(BAND_CODE);
    modelService.attach(band);
    band.setName(BAND_NAME);
    band.setAlbumSales(ALBUMS SOLD);
    band.setHistory(BAND_HISTORY); return
    группа;
}
@До
публичная пустота настраивать()
{
    // Мы будем тестировать POJO DefaultBandFacade — реализацию интерфейса BandFacade. bandFacade = new DefaultBandFacade();

    модельСервис = mock(ModelService.класс);
    bandService =
    // Затем мы подключаем эту службу к реализации BandFacade.
    bandFacade.setBandService(bandService);
}
/**
 * Целью данного теста является проверка того, что:
 *
 * 1) Метод фасада getBands вызывает метод BandService getBands
 *
 * 2) Затем фасад правильно обворачивает BandModels, которые возвращаются ему из getBands BandService, в Data
 * Передача объектов типа BandData.
 */
@Тест
общественный недействительный testGetAllBands () {

    /**
     * Мы создаем экземпляр объекта, который мы хотели бы вернуть в BandFacade, когда имитированный BandService
     * вызывается метод getBands. Это будет список из двух BandModels.
     */
    final List<BandModel> bands = dummyDataBandList(); // создаем
    тестовую полосу для сравнения утверждений final BandModel
    band = configTestBand();
    // Мы сообщаем Mockito, что ожидаем вызова метода BandService getBands, и что когда это произойдет, должны быть возвращены полосы
    when(bandService.getBands()).thenReturn(bands);
    /**
     * Теперь мы вызываем getBands BandFacade. Если в этом методе вызывается getBands BandService,
     * Mockito вернет ему экземпляр bands. Mockito также запомнит, что вызов был сделан.
     */
    final List<BandData> dto = bandFacade.getBands(); // Теперь мы
    проверяем, является ли dto DTO-версией bands.
    Assert.assertNotNull(dto);
    Assert.assertEquals(bands.size(), dto.size());
    Assert.assertEquals(band.getCode(), dto.get(0).getId());
    Assert.assertEquals(band.getName(), Assert.assertEquals(band.getAlbumSales(), Assert.assertEquals(band.getDescription(), dto.get(0).getAlbumsSold()));


}
@Тест
public void testGetBand() {

    // создаем тестовую полосу
    окончательная полоса BandModel = configTestBand();
    // Мы сообщаем Mockito, что ожидаем вызова метода BandService getBandForCode, и что когда это произойдет, тестовая полоса должна быть возвращена
    when(bandService.getBandForCode(BAND_CODE)).thenReturn(band);
    окончательный BandData dto = bandFacade.getBand(BAND_CODE);
    // Теперь мы проверяем, является ли полоса правильным DTO-представлением тестовой модели полосы
    Assert.assertNotNull(dto);
    Assert.assertEquals(band.getCode(), Assert.assertEquals(band.getName(),
    Assert.assertEquals(band.getAlbumSales(), Assert.assertEquals(band.getDescription(), dto.get(0).getAlbumsSold()));


}
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать модульный тест самостоятельно или хотите пропустить этот шаг, скопируйте

<HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/facades/impl/DefaultBandFacadeUnitTest.java <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/testsrc/concerttours/facades/impl/каталог.

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/facades/impl/DefaultBandFacadeUnitTest.java \$HYBRI

Окна

12/3/24, 11:01 утра

```
echo f | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\tests\src\concerttours\facades\impl\DefaultBandFacadeUnitTest.j
```

. Перестройте SAP Commerce с помощью Ant.

```
cd $HYBRIS_HOME_DIR\hybris\bin\platform; ant очистить все
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и очистите все
```

. Запустите тесты и убедитесь, что они пройдены.

```
hybris@ всетести - Dtestclasses.packages="concerttours.*"
```

```
hybris@ всетести - Dtestclasses.packages=концертные туры.*
```

. Посмотреть результаты теста можно на <*%HYBRIS\_HOME\_DIR%*>/hybris/log/junit/index.html подтвердите, что вы видите тест интеграции BandFacadeIntegrationTest.

. Запустите testFacadeLayerOk повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR\hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testFacadeLayerOk тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testFacadeLayerOk тест
```

. Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR\hybris; git add . ; git commit -m "Расширить слой фасада"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Расширить слой фасада"
```

## Передняя часть

После того, как у вас есть модель и бизнес-логика, вы можете разработать подходящее веб-приложение front-end. При создании front-end используйте фреймворк Spring MVC для разделения частей модели, представления и контроллера.

SAP Commerce предоставляет ряд витрин Accelerator. Accelerator предоставляет вам базовые строительные блоки, которые вы можете использовать для разработки сложных и адаптивных витрин в определенных коммерческих доменах. Но чтобы проиллюстрировать основы того, как строятся магазины, вы сосредоточитесь здесь на двух аспектах.

Класс контроллера

Интерфейс между действием пользователя и базовой моделью.

Страницы JSP

Динамические веб-страницы для представления данных конечному пользователю.

Ранее вы создали модельный аспект вашей архитектуры MVC, теперь вы создаете представление, страницы JSP и контроллер, класс контроллера.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Фасадный слой](#)

Следующий:[Динамические Атрибуты](#)

## Создание компонента веб-приложения с использованием Spring MVC

Разработайте простой интерфейсный магазин и получите основы понимания того, как устроены магазины.

## Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.java:

```
public void testWebAppComponent() выдает исключение
{ canLoginToHybrisCommerce();
navigationTo("https://localhost:9002/concerttours/bands"); waitFor("a", "The
Quiet");
navigationTo("https://localhost:9002/concerttours/bands/A007");
assertTrue(waitFor("p", "Английское хоровое общество, специализирующееся на прекрасно аранжированных, успокаивающих мелодиях и песнях")); }
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR\hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testWebAppComponent test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testWebAppComponent test
```

## Процедура

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR\hybris\bin\platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Создайте два новых класса контроллеров с именами concerttours.controller.BandController и концерттурс.контроллер.TourController под веб/источник папка в концертные туры расширение.

```
упаковка concerttours.controller;
импорт de.hybris.platform.catalog.CatalogVersionService;
импорт java.io.UnsupportedEncodingException;
импорт java.net.URLDecoder;
импорт java.util.List;
импорт org.springframework.beans.factory.annotation.Autowired;
импорт org.springframework.stereotype.Controller;
импорт org.springframework.ui.Model;
импорт org.springframework.web.bind.annotation.PathVariable;
импорт org.springframework.web.bind.annotation.RequestMapping;
импорт concerttours.data.BandData;
импорт концертные туры.фасады.BandFacade;

@Controller
публичный класс BandController
{
    частная статическая окончательная строка CATALOG_ID = "concertToursProductCatalog";
    частная статическая окончательная строка CATALOG_VERSION_NAME = "Online"; частная
        CatalogVersionService catalogVersionService; BandFacade
    частный bandFacade;
    @RequestMapping(значение = "/полосы")
    публичная строка showBands(окончательная модель model) {

        окончательный список<BandData> полосы = bandFacade.getBands();
        model.addAttribute("полосы", полосы);
        вернуть "BandList";
    }
    @RequestMapping(значение = "/bands/{bandId}")
    public String showBandDetails(@PathVariable final String bandId, final Model model) выдает исключение UnsupportedEncodingException {

        catalogVersionService.setSessionCatalogVersion(CATALOG_ID, CATALOG_VERSION_NAME); финальная строка
        decodedBandId = URLDecoder.decode(bandId, "UTF-8");
        окончательный диапазон BandData = bandFacade.getBand(decodedBandId);
        model.addAttribute("полоса", полоса);
        вернуть "BandDetails";
    }
    @Autowired
    public void setCatalogVersionService(final CatalogVersionService catalogVersionService) {

        this.catalogVersionService = catalogVersionService;
    }
    @Autowired
    public void setFacade(финальный фасад BandFacade) {

        эта.bandFacade = фасад;
    }
}

упаковка concerttours.controller;
импорт de.hybris.platform.catalog.CatalogVersionService;
импорт java.io.UnsupportedEncodingException;
импорт java.net.URLDecoder;
импорт org.springframework.beans.factory.annotation.Autowired;
импорт org.springframework.stereotype.Controller;
импорт org.springframework.ui.Model;
импорт org.springframework.web.bind.annotation.PathVariable;
импорт org.springframework.web.bind.annotation.RequestMapping;
импорт concerttours.data.TourData;
импорт концерттуры.фасады.TурФасад;

@Controller
публичный класс ТурКонтроллер
{
    частная статическая окончательная строка CATALOG_ID = "concertToursProductCatalog";
    частная статическая окончательная строка CATALOG_VERSION_NAME = "Online"; частная
        КаталогВерсияСервис каталогВерсияСервис; ТурФасад
    частный турФасад;
    @RequestMapping(значение = "/tours/{tourId}")
    public String showTourDetails(@PathVariable final String tourId, final Model model) выдает исключение UnsupportedEncodingException {

        catalogVersionService.setSessionCatalogVersion(CATALOG_ID, CATALOG_VERSION_NAME); финальная строка
        decodedTourId = URLDecoder.decode(tourId, "UTF-8");
        окончательный TourData тип = tourFacade.getTourDetails(decodedTourId);
        model.addAttribute("тип", тип);
        вернуть «TourDetails»;
    }
    @Autowired
    public void setCatalogVersionService(final CatalogVersionService catalogVersionService) {

        this.catalogVersionService = catalogVersionService;
    }
    @Autowired
    public void setFacade(финальный фасад TourFacade) {

        этот.турФасад = фасад;
    }
}
```

В этих классах вы используете несколько аннотаций. @Контроллер аннотация сообщает Spring, что созданный класс следует обрабатывать как компонент Spring Controller.

В BandController, вы можете увидеть два метода, которые аннотированы с помощью @ЗапросКартографирования аннотации. Это отображает пути /полосы и /полоса/{bandCode}. Контроллер методы. В то время как первый метод showBands Метод прост, showBandDetails Метод также объявляет @Переменная\_пути. The bandId параметр автоматически разрешается Spring. Оба метода возвращают простые строки, которые будут подхвачены ViewResolver, которые заявили. Возвращенная строка приветствуя этому будет сопоставлено с WEB-INF/views/hello.jsp.

Чтобы четко отделить часть View от Model, заполните переданный объект Model данными, которые вы хотели бы получить в представлении. Либо List of BandData или один BandData экземпляр будет частью модели, переданной в представление.

Вам необходимо задать каталог сеанса, поскольку гибкий поиск ограничивает любой поиск элементов, поддерживающих каталог, каталогами, перечисленными в объекте сеанса.

The ТурКонтроллер класс кодируется аналогично.

12/3/24, 11:01 утра

Интерактивный ярлык SAP Commerce 123: если вы не можете создать классы или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/controllerrc содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/controller.

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/controller/BandController.java \$HYBRIS\_HOME\_DIR/hybris

эхо д | xcopy %HYBRIS\_HOME\_DIR%hybris123\src\main\webapp\resources\concerttours\src\concerttours\controller %HYBRIS\_HOME\_DIR%\hybris\bin\cust

. Создайте BandList.jsp, BandDetails.jsp, и Подробности тура.jsp для веб/webroot/WEB-INF/просмотра

```
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %> <!doctype html>

<html>
    <title>Группа Список</title>
    <тело>
        <h1>Группа Список</h1>
        <уна>
            <c:forEach var="band" items="${bands}">
                <li><a href="#">\${band.name}</a></li>
            </c:forEach>
            </ul>
        </тело>
    </html>

<%@taglib префикс="c" uri="http://java.sun.com/jsp/jstl/core"%>
<тип документа html>
<html>
<title>Группа Подробности</title>
<тело>
    <h1>Группа Подробности</h1>
    Подробности о группе ${band.name}
    <p>${band.description}</p> <p>Тип
    музыки:</p>
    <ул>
        <c:forEach var="жанр" элементы="${band.genres}">
            <li>${жанр}</li>
        </c:forEach>
    </ул>
    <p>История тура:</p>
    <ул>
        <c:forEach var="тур" элементы="${band.tours}">
            <li><a href="#">\${tour.tourName}</a>(количество концертов: ${tour.numberOfConcerts})</li> </c:forEach>
        </ul>
        <a href="#">..//bands>Вернуться к списку групп</a>
    </тело>
</html>

<%@taglib префикс="c" uri="http://java.sun.com/jsp/jstl/core"%> <%@taglib uri="http://java.sun.com/jsp/jstl/fmt" префикс="fmt" %> <!doctype html>

<html>
<title>Тур Подробности</title>
<тело>
    <h1>Тур Подробности</h1>
    Подробности тура для ${tour.tourName}
    <p>${tour.description}</p>
    <p>Расписание:</p>
    <таблица>
        <тр><th>Место проведения</th><th>Дата</th></tr>
        <c:forEach var="concert" items="${tour.concerts}">
            <тр><td>${concert.venue}</td><td>${concert.type}</td><td><fmt:formatDate pattern="dd MMM yyyy" value="${concert.date}" /></td></tr> </c:forEach>
        </таблица>
        <a href="#">..//bands>Вернуться к списку групп</a>
    </тело>
</html>
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать страницы списка и JSP или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/web/webroot/WEB-INF/viewsc содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/web/webroot/WEB-INF/views.

mkdir -p \$HYBRIS\_HOME\_DIR/hybris/bin/custom/concerttours/web/webroot/WEB-INF/views; cp \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/con

echo d | xcopy %HYBRIS\_HOME\_DIR%hybris123\src\main\webapp\resources\concerttours\web\weboot\WEB-INF\views %HYBRIS\_HOME\_DIR%\hybris\bin\custom

. Перестройте SAP Commerce с помощью Ant.

cd \$HYBRIS\_HOME\_DIR/hybris/bin/platform; ant очистить все

cd %HYBRIS\_HOME\_DIR%\hybris\bin\platform и очистите все

. Запустите SAP Commerce.

cd \$HYBRIS\_HOME\_DIR/hybris/bin/platform; ./hybrisserver.sh start

cd %HYBRIS\_HOME\_DIR%\hybris\bin\platform и запустите hybrisserver.bat

. Перейдите к <https://localhost:9002/concerttours/bands> и просмотреть список групп. Теперь вы можете просмотреть список групп, а также подробную информацию об отдельных группах и подробности их туров. Обратите внимание, что вам не нужно перезапускать сервер, если вы создаете новые или изменяете существующие страницы jsp. Это может быть очень полезно для ускорения разработки вашего интерфейса.

. Запустите testWebAppComponent и проверьте приемочное испытание и подтвердите, что оно теперь прошло.

cd \$HYBRIS\_HOME\_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testWebAppComponent test

cd %HYBRIS\_HOME\_DIR%hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testWebAppComponent test

. Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add . ; git commit -m "Создание компонента веб-приложения с использованием Spring MVC"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Создание компонента веб-приложения с использованием Spring MVC"
```

## Динамические Атрибуты

Динамические атрибуты позволяют добавлять атрибуты в модель и создавать для них пользовательскую логику, не касаясь самого класса модели. Они предоставляют способ генерировать новые данные и получать к ним доступ без вызова отдельной службы для этого. Динамические атрибуты — это временные данные, которые не сохраняются в базе данных.

Все атрибуты, которые вы определили для своего ГруппыЭлементы просто сохраняются и извлекаются без какой-либо дополнительной бизнес-логики. Вы используете методы getter и setter соответствующего класса модели Java для доступа к ним и их изменения. Однако вы также можете динамически вычислять значения для элемента и раскрывать эти значения с помощью методов getter и setter. Существует много причин, по которым вы можете захотеть сделать это:

- Вы можете захотеть представить значение в другом наборе единиц. Например, вам может потребоваться представить точку на карте как в полярных, так и в декартовых координатах, время в 12-часовом и 24-часовом формате или относительную и абсолютную версию пути к каталогу или URL.
- Вам может понадобиться рассчитать значение из нескольких атрибутов элемента. Например, вам может потребоваться рассчитать расстояние между двумя атрибутами, содержащими местоположение, сложить значения ряда атрибутов, таких как цена, налог и скидка в элементе заказа, чтобы получить промежуточный итог, или определить возраст человека из атрибута даты рождения и текущей даты.

Чтобы предоставить пользовательскую логику для динамического атрибута типа элемента, вам необходимо предоставить логику чтения и записи в зависимости от требований к этому атрибуту. Вы не можете добавить эту логику непосредственно в классы модели, потому что они уже сгенерированы, и все, что вы добавите, будет перезаписано и потерянно при следующей перестройке системы. Вместо этого напишите простой класс Java, который реализует DynamicAttributeHandlerинтерфейс. Этот интерфейс использует дженерики Java, так что фактический класс отражает как тип значения атрибута (в данном случаеjava.lang.Dлинный) и класс модели, с которым он работает (в данном случаеКонцерт). Затем вы объявляете этот класс обработчика как компонент Spring.

Для этого следа вы создаете новый атрибут под названием Концерт.дниОсталось, который показывает динамически рассчитанное количество дней, оставшихся до концерта. Это значение затем можно использовать для отображения обратного отсчета до даты концерта на веб-странице, например.

Реализация этого нового динамического атрибута имеет несколько ключевых особенностей:

- Тип сохранения установлен на динамический
- НастойчивостьattributeHandlerуказывает на компонент, который должен обрабатыватьDynamicAttributeHandlerинтерфейс
- Theписатьатрибут установлен на значение false, и поэтому атрибут доступен только для чтения

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Передняя часть](#)

Следующий:[Интеграция динамических атрибутов](#)

## Ввести динамический атрибут

Создайте Концерт.дниОсталось динамический атрибут.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.java на языке.

```
public void testDynamicAttributeUnitTest() {
    assertEquals(0, checkTestSuiteXMLMatches("(.*)testsuite ошибки=\"0\" неудачи=\"0\" (.*) имя=\"ConcertDaysUntilAttributeHandlerUnitTest\" пакет=\"conc\""));
}
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testDynamicAttributeUnitTest тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testDynamicAttributeUnitTest тест
```

### Процедура

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Определите новый динамический обработчик атрибутов в элементе Concert в <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttours-items.xml ле.

```
<квалификатор атрибута="daysUntil" тип="java.lang.Long">
    <persistence type="dynamic" attributeHandler="concertDaysUntilAttributeHandler" /> <modifiers read="true"
    write="false" />
</атрибут>
```

Интерактивный ярлык SAP Commerce 123: если вы не можете самостоятельно определить новый динамический атрибут или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttours-items.xml содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-itemsWithDynamicAttributes.xml

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-itemsWithDynamicAttributes.xml $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-itemsWithDynamicAttributes.xml
```

копировать /у %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\resources\concerttours-itemsWithDynamicAttributes.xml %HYBRIS\_HOME\_D

. Определите обработчик атрибутов в  
 <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/attributehandlers/ConcertDaysUntilAttributeHandler.java

```
упаковка concerttours.attributehandlers;
импорт de.hybris.platform.servicelayer.model.attribute.AbstractDynamicAttributeHandler; java.time.Duration;
импорт импорт java.time.ZoneId;
импорт java.time.ZonedDateTime;
импорт org.springframework.stereotype.Component;
импорт concerttours.model.ConcertModel;

@Компонент
открытый класс ConcertDaysUntilAttributeHandler расширяет AbstractDynamicAttributeHandler<Long, ConcertModel> {

  @Переопределить
  public Long get(final ConcertModel model) {

    если (модель.getDate() == null) {
      вернуть ноль;
    }
    final ZonedDateTime concertDate = model.getDate().toInstant().atZone(ZoneId.systemDefault()); final ZonedDateTime now =
    ZonedDateTime.now();
    если (концертДата.isBefore(сейчас)) {
      вернуть Long.valueOf(0L);
    }
    окончательная Продолжительность продолжительность =
    Продолжительность.between(now, concertDate); return Long.valueOf(duration.toDays());
  }
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете определить обработчик атрибутов самостоятельно или хотите пропустить этот шаг, замените  
 <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/attributehandlers/ConcertDaysUntilAttributeHandler.javac содержанием

<HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/attributehandlers/ConcertDaysUntilAttributeHandler.

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/attributehandlers/ConcertDaysUntilAttributeHandler.jav

echo f | xcopy %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\attributehandlers\ConcertDaysUntilAttributeH

. Зарегистрируйте обработчик атрибутов в Spring в <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttours-spring.xml

```
<bean id="concertDaysUntilAttributeHandler" class="concerttours.attributehandlers.ConcertDaysUntilAttributeHandler"/>
```

Интерактивный ярлык SAP Commerce 123: если вы не можете определить обработчик атрибутов самостоятельно или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttours-spring.xmllc содержанием  
 <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-spring-withAttributeHandler.xml

cp \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-spring-withAttributeHandler.xml \$HYBRIS\_HOME\_DIR/hy

копировать /у %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\resources\concerttours-spring-withAttributeHandler.xml %HYBRIS\_HOME\_

. Создайте модульный тест для подтверждения правильного поведения в  
 <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/testsrc/concerttours/attributehandlers/ConcertDaysUntilAttributeHandlerUnitTests.java

```
упаковка concerttours.attributehandlers;
импорт java.util.Date;
импорт org.junit.Утверждение;
импорт org.junit.Test;
импорт concerttours.model.ConcertModel;
импорт de.hybris.bootstrap.annotations.UnitTest;

@UnitTest
публичный класс ConcertDaysUntilAttributeHandlerUnitTests {
  @Тест
  public void testGetFutureConcertDate() выдает исключение {
    финальный концерт ConcertModel = new ConcertModel();
    окончательный обработчик ConcertDaysUntilAttributeHandler = new ConcertDaysUntilAttributeHandler();
    окончательная Дата futureDate = new Date(new Date().getTime() + 49 * 60 * 60 * 1000); concert.setDate(futureDate);

    Assert.assertEquals("Неверное значение для концерта в будущем", 2L, handler.get(concert).longValue());
  }
  @Тест
  общественный недействительный testGetNullConcertDate () {
    финальный концерт ConcertModel = new ConcertModel();
    окончательный обработчик ConcertDaysUntilAttributeHandler = new ConcertDaysUntilAttributeHandler();
    Assert.assertNull(handler.get(concert));
  }
  @Тест
  public void testGetPastConcertDate() выдает исключение {
    финальный концерт ConcertModel = new ConcertModel();
    окончательный обработчик ConcertDaysUntilAttributeHandler = new ConcertDaysUntilAttributeHandler();
    окончательная Дата futureDate = new Date(new Date().getTime() - 24 * 60 * 60 * 1000); concert.setDate(futureDate);

    Assert.assertEquals("Неверное значение для концерта в прошлом", 0L, handler.get(concert).longValue());
  }
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать модульный тест самостоятельно или хотите пропустить этот шаг, замените

<HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/testsrc/concerttours/attributehandlers/ConcertDaysUntilAttributeHandlerUnitTests.java с содержанием

<HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/attributehandlers/ConcertDaysUntilAttributeHand

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/attributehandlers/ConcertDaysUntilAttributeHandler

echo f | xcopy %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\testsrc\concerttours\attributehandlers\ConcertDaysUntilAttributeHandler

.Перестройте SAP Commerce с помощью Ant.

cd \$HYBRIS\_HOME\_DIR/hybris/bin/platform; ant очистить все

cd %HYBRIS\_HOME\_DIR%\hybris\bin\platform и очистите все

.Обновление SAP Commerce с помощью Ant.

cd \$HYBRIS\_HOME\_DIR/hybris/bin/platform; ant обновление системы

cd %HYBRIS\_HOME\_DIR%\hybris\bin\platform и ant updatesystem

.Запустите модульный тест.

cd \$HYBRIS\_HOME\_DIR/hybris/bin/platform; ant unittests -Dtestclasses.packages="concerttours.\*"

cd %HYBRIS\_HOME\_DIR%\hybris\bin\platform и ant unittests -Dtestclasses.packages=concerttours.\*

.Посмотреть результаты теста можно на <HYBRIS\_HOME\_DIR>/hybris/log/junit/index.html подтвердите, что этот модульный тест пройден.

.запустите testDynamicAttributeUnitTest повторите приемочное испытание и подтвердите, что оно теперь пройдено.

cd \$HYBRIS\_HOME\_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testDynamicAttributeUnitTest тест

cd %HYBRIS\_HOME\_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testDynamicAttributeUnitTest тест

.Зафиксируйте изменения в локальном репозитории Git.

cd \$HYBRIS\_HOME\_DIR/hybris; git add . ; git commit -m "Ввести динамический атрибут"

cd %HYBRIS\_HOME\_DIR%\hybris & git add . & git commit -m "Ввести динамический атрибут"

## Интеграция динамических атрибутов

Хорошей практикой является всегда проводить интеграционное тестирование при внедрении новых функций, чтобы тестировать новые классы в контексте.

Ранее вы запустили модульный тест для проверки базовой логики вашего нового класса динамических атрибутов. Теперь напишите интеграционный тест, чтобы проверить его. Вы хотите предоставить классу некоторые данные в реальном времени, а также проверить надежность класса с будущими, прошлыми и пустыми значениями даты. Используйте модель Сервис, чтобы гарантировать, что результат будет иметь форму расширенной модели элемента, как и ожидалось.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Динамические Атрибуты](#)

Следующий:[Обновление веб-страницы](#)

## Тестирование интеграции динамических атрибутов

Создайте и запустите интеграционный тест, чтобы убедиться, что обработчик правильно выбран как компонент Spring.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест Hybris123Tests.java.

```
public void testDynamicAttributeIntegrationTest() {
    assertTrue(checkTestSuiteXMLMatches("(.* )testsuite ошибки=""0"" неудачи=""0"" (.*) имя=""ConcertDaysUntilAttributeHandlerIntegrationTest"" пакет }
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

cd \$HYBRIS\_HOME\_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testDynamicAttributeIntegrationTest тест

cd %HYBRIS\_HOME\_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testDynamicAttributeIntegrationTest тест

## Процедура

.Остановите SAP Commerce.

cd \$HYBRIS\_HOME\_DIR/hybris/bin/platform; ./hybrisserver.sh остановить

cd %HYBRIS\_HOME\_DIR%\hybris\bin\platform и hybrisserver.bat остановить

.Создайте интеграционный тест:

```
упаковка concerttours.attributehandlers;
импорт de.hybris.bootstrap.annotations.IntegrationTest;
```

12/3/24, 11:01 утра

```
импорт de.hybris.platform.servicelayer.ServicelayerTransactionalTest;
импорт de.hybris.platform.servicelayer.model.ModelService; java.util.Date;
импорт
импорт javax.annotation.Resource;
импорт org.junit.Assert;
импорт org.junit.Test;
импорт концертные туры.модель.ConcertModel;

@Интеграционный тест
открытый класс ConcertDaysUntilAttributeHandlerIntegrationTest расширяет ServicelayerTransactionalTest {

    @Ресурс
    частный ModelService modelService; @Test

    public void testGetFutureConcertDate() выдает исключение {

        окончательная ConcertModel концерт = modelService.create(ConcertModel.class); окончательная Дата
        futureDate = new Date(new Date().getTime() + 49 * 60 * 60 * 1000); концерт.setDate(futureDate);

        Assert.assertEquals("Неверное значение для концерта в будущем: " + concert.getDaysUntil().longValue(), 2L, concert.getDaysUntil().longValue)
    }
    @Тест
    общественный недействительный testGetNullConcertDate () {

        финальный ConcertModel концерт = modelService.create(ConcertModel.class);
        Assert.assertNull("Нет даты концерта, которая не возвращает null: "+ concert.getDaysUntil(), concert.getDaysUntil());
    }
    @Тест
    public void testGetPastConcertDate() выдает исключение {

        окончательная ConcertModel концерт = modelService.create(ConcertModel.class); окончательная
        Дата pastDate = new Date(new Date().getTime() - 24 * 60 * 60 * 1000); концерт.setDate(pastDate);

        Assert.assertEquals("Неверное значение для концерта в прошлом: "+concert.getDaysUntil().longValue(), 0L, concert.getDaysUntil().longValue())
    }
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать тестовый класс самостоятельно или хотите пропустить этот шаг, замените  
<HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/testsrc/concerttours/attributehandlers/ConcertDaysUntilAttributeHandlerIntegrationTest с содержанием

<HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/attributehandlers/ConcertDaysUntilAttributeHandler

cp \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/attributehandlers/ConcertDaysUntilAttributeHandlerInt

копировать /у %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\testsrc\concerttours\attributehandlers\ConcertDaysUntilAttributeHand

.Перестройте SAP Commerce с помощью Ant.

cd \$HYBRIS\_HOME\_DIR/hybris/bin/platform; ant очистить все

cd %HYBRIS\_HOME\_DIR%\hybris\bin\platform и очистите все

.Инициализируйте своего тестового арендатора.

cd \$HYBRIS\_HOME\_DIR/hybris/bin/platform; муравей юнитинит

cd %HYBRIS\_HOME\_DIR%\hybris\bin\platform и ant yunitinit

.Запустите интеграционные тесты и убедитесь, что оба новых интеграционных теста пройдены.

муравей интеграционныетести - Dtestclasses.packages="concerttours.\*"

муравей интеграционныетести - Dtestclasses.packages=концертные туры.\*

.Посмотреть результаты теста можно на <HYBRIS\_HOME\_DIR>/hybris/log/junit/index.html подв ConcertDaysUntilAttributeHandlerIntegrationTest прошло.

.Запустите тестДинамическийАтрибутИнтеграцияТест повторите приемочное испытание и подтвердите, что оно теперь пройдено.

cd \$HYBRIS\_HOME\_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testDynamicAttributeIntegrationTest тест

cd %HYBRIS\_HOME\_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testDynamicAttributeIntegrationTest тест

.Зафиксируйте изменения в локальном репозитории Git.

cd \$HYBRIS\_HOME\_DIR/hybris; git add . ; git commit -m "Проверка интеграции динамических атрибутов"

cd %HYBRIS\_HOME\_DIR%\hybris & git add . & git commit -m "Тестирование интеграции динамических атрибутов"

## Обновление веб-страницы

Обновите соответствующие части вашего расширения, чтобы использовать новый динамический атрибут.

После введения новых атрибутов элемента вы хотите применить их. В этом случае вы хотите отобразить количество дней, оставшихся до концерта, на странице сведений о туре. Для этого вам нужно обновить TourFacade, TourDetails.jsp страница, и КонцертSummaryData класс, чтобы вычисленное значение динамического атрибута Концерты.дниОсталось появляется в передней части.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Интеграция динамических атрибутов](#)

Следующий:[События и слушатели](#)

## Обновите веб-страницу, включив новый динамический атрибут

Добавить обработку для Концерты.дниОсталось атрибут и отобразите его на странице сведений о туре вашего веб-приложения.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.java.

```
public void testDynamicAttributeView() выдает исключение
{ canLoginToHybrisCommerce();
navigationTo("https://localhost:9002/concerttours/bands/A001"); waitFor("a","The
Grand Little x Tour");
navigationTo("https://localhost:9002/concerttours/tours/201701"); waitFor("th","Дней до");

assertTrue(waitFor("td","0")); }
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testDynamicAttributeView test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testDynamicAttributeView test
```

### Процедура

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Обновите определение компонента, включив в него новый динамический атрибут.concerttours-beans.xml.

```
<класс компонента = "concerttours.data.ConcertSummaryData">
    <description>Объект данных для краткого содержания концерта</description>
    <property name = "id" тип = "String" />
    <имя свойства = "date" тип = "java.util.Date" /> <имя свойства =
    "venue" тип = "String" /> <имя свойства = "type" тип = "String" />
    <имя свойства = "countDown" тип="Long"/>

</бо6>
```

Интерактивный ярлык SAP Commerce 123: если вы не можете обновить определение компонента самостоятельно или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttours-beans.xml содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-beansWithDynamicAttributes.xml

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-beansWithDynamicAttributes.xml $HYBRIS_HOME_DIR/hyb
```

```
копировать /у %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\resources\concerttours-beansWithDynamicAttributes.xml %HYBRIS_HOME_D
```

. Обновите реализацию фасада тура по умолчанию, включив в нее настройку нового динамического атрибута.

```
упаковка concerttours.facades.impl;
импорт de.hybris.platform.core.model.product.ProductModel;
импорт de.hybris.platform.product.ProductService;
импорт de.hybris.platform.variants.model.VariantProductModel; java.util.ArrayList;
импорт
импорт java.util.List;
импорт org.springframework.beans.factory.annotation.Required;
импорт concerttours.data.ConcertSummaryData;
импорт concerttours.data.Данные тура;
импорт concerttours.enums.Тип концерта;
импорт concerttours.фасады.Фасад тура;
импорт concerttours.модель.Модель концерта;

открытый класс DefaultTourFacade реализует TourFacade {
    частный ProductService productService; @Override
    public TourData getTourDetails(final String tourId) {
        если (tourId == null) {
            throw new IllegalArgumentException("Идентификатор тура не может быть пустым");
        }
        конечный ProductModel продукт = productService.getProductForCode(tourId); если (продукт ==
        null)
        {
            вернуть ноль;
        }

        // Создаем список ConcertSummaryData из совпадений
        окончательный список<ConcertSummaryData> концерты = new ArrayList<>();
        если (product.getVariants() != null)
        {
            для (окончательный VariantProductModel : product.getVariants()) {
                если (вариант instanceof ConcertModel) {
                    финал ConcertModel концерт = (ConcertModel) вариант; финал
                    ConcertSummaryData резюме = new ConcertSummaryData();
                    summary.setId(concert.getCode());
                }
            }
        }
    }
}
```

```

summary.установитьДат(концерт.получитьДат()); summary.установитьМесто
проводения(концерт.получитьМестоПроведения());
summary.setTipo(концерт.getConcertType() == ConcertType.OPENAIR ? "На открытом воздухе" : "В помещении");
summary.setCountDown(концерт.getDaysUntil());
концерты.добавить(рэзюме);
}

}

// Теперь мы можем создать объект передачи TourData final
TourData tourData = new TourData();
tourData.setId(product.getCode());
tourData.setTourName(product.getName());
tourData.setDescription(product.getDescription());
tourData.setConcerts(концерты);
вернуть данные тура;
}

@Необходимый
public void setProductService(final ProductService productService) {

    этот.productService = productService;
}
}

```

Интерактивный ярлык SAP Commerce 123: если вы не можете обновить реализацию фасада тура по умолчанию самостоятельно или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/facades/impl/DefaultTourFacade.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/facades/impl/DefaultTourFacadeWithDynamicAttributes

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/facades/impl/DefaultTourFacadeWithDynamicAttributes.java
```

```
копировать /у %HYBRIS_HOME_DIR%\\hybris123\\src\\main\\webapp\\resources\\concerttours\\src\\concerttours\\facades\\impl\\DefaultTourFacadeWithDynamicAttributes
```

.Обновите страницу просмотра.

```
<%@taglib префикс="c" uri="http://java.sun.com/jsp/jstl/core"%> <%@taglib uri="http://
java.sun.com/jsp/jstl/fmt" префикс="fmt" %> <!doctype html>

<html>
<title>Тур      Подробности</title>
<тело>
    <h1>Тур      Подробности</h1>
    Подробности тура для ${tour.tourName}
    <p>${tour.description}</p>
    <p>Расписание:</p>
    <таблица>
        <tr><th>Место проведения</th><th></th><th>Дата</th><th>Дни   Пока</th></tr>
        <c:forEach var="concert" items="${tour.concerts}">
            <tr><td>${concert.type}</td><td>${concert.venue}</td><td>${concert.date}</td><td>${concert.type}</td></tr>
        </c:forEach>
    </таблица>
    <a href="../bands">Вернуться к списку групп</a>
</body>
</html>
```

Интерактивный ярлык SAP Commerce 123: если вы не можете обновить страницу просмотра самостоятельно или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/web/webroot/WEB-INF/views/TourDetails.jspc содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/web/webroot/WEB-INF/views/TourDetailsWithDynamicAttributes.jsp

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/web/webroot/WEB-INF/views/TourDetailsWithDynamicAttributes.jsp $HYBRIS_HOME
```

```
копировать /у %HYBRIS_HOME_DIR%\\hybris123\\src\\main\\webapp\\resources\\concerttours\\web\\webroot\\WEB-INF\\views\\TourDetailsWithDynamicAttributes.jsp %HYBRIS_HOME
```

.Перестройте SAP Commerce с помощью Ant.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ant очистить все
```

```
cd %HYBRIS_HOME_DIR%\\hybris\\bin\\platform и очистите все
```

.Запустите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh start
```

```
cd %HYBRIS_HOME_DIR%\\hybris\\bin\\platform и запустить hybrisserver.bat
```

.Перейдите к <https://localhost:9002/concerttours/bands> и просмотреть список групп.

Каждый тур теперь показывает дней дозначение, которое указывает количество дней до начала тура.

.Запустите `testDynamicAttributeView` повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testDynamicAttributeView test
```

```
cd %HYBRIS_HOME_DIR%\\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testDynamicAttributeView test
```

.Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add .; git commit -m "Обновить веб-страницу для включения нового динамического атрибута"
```

```
cd %HYBRIS_HOME_DIR%\\hybris & git add . & git commit -m "Обновить веб-страницу для включения нового динамического атрибута"
```

## События и слушатели

Система событий — это структура, предоставляемая `ServiceLayer`, который позволяет отправлять и получать события в SAP Commerce.

Вы можете настроить компоненты вашего расширения для публикации событий, которые затем будут получены зарегистрированными слушателями. Слушатели — это объекты, которые уведомляются о событиях и выполняют бизнес-логику в зависимости от произошедшего события. События могут быть опубликованы локально или в кластере узлов. Вы можете зарегистрировать новых слушателей как Spring-бины в вашем XML-файле конфигурации Spring.

Платформа определяет и публикует события для ряда предопределенных типов событий. Они включают в себя AfterItemCreationEvent тип, элементы которого публикуются после сохранения любого нового элемента данных в базе данных. Для обработки этих AfterItemCreationEvent события, вы предоставляете класс прослушивателя и регистрируете его в фреймворке событий.

Для вашегоконцертные турырасширение, вы хотите генерировать новости, которые вы можете потенциально отправлять зарегистрированным подписчикам через различные каналы. Вы хотите создать новый Новостиэлемент всякий раз, когда новая группа подписана. В вашем классе слушателя вы определяете onEvent метод для указания кода, который вы хотите выполнить при возникновении этого события. В этом случае вы проверяете, является ли новый элемент полосой. Если это так, вы просите службу модели платформы создать и сохранить новый элемент новостей.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Обновление веб-страницы](#)

Следующий:[Перехватчики и пользовательские события](#)

## Сопутствующая информация

[Система событий](#)

## Создать слушателя

Напишите класс слушателя, который создает новый Новостиэлемент, когда новый Группаэлемент создается и сохраняется в базе данных.

## Предпосылки

В ваш комплект документов включен следующий приемочный тест Hybris123Tests.java.

```
public void testNewsEvents() { canLoginToHybrisCommerce();
navigationTo("https://localhost:9002/platform/init");
waitForThenClickButtonWithText("Инициализировать");
waitForThenClickOkInAlertWindow();

waitForInitToComplete();
закрытьБраузер();

canLoginToHybrisCommerce(); navigationTo("https://localhost:9002/console/flexsearch");
waitForFlexQueryFieldThenSubmit("ВЫБРАТЬ {заголовок} ИЗ {новостей}");
assertTrue(waitFor("td", "Новая группа, запрещена"));

}
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testNewsEvents test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testNewsEvents test
```

## Процедура

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Добавьте новый тип элемента под названием Новости в concerttours-items.xml

```
<itemtype generate="true" code="News" autocreate="true"> <deployment
    table="News" typecode="30270" />
<атрибуты>
    <квалификатор атрибута="дата" тип="java.util.Date">
        <description>дата новости</description> <persistence
            type="property"
        />
    </атрибут>
    <квалификатор атрибута="заголовок"      тип="java.lang.String">
        <description>краткий заголовок для новости</description> <persistence
            type="property"
        />
    </атрибут>
    <квалификатор атрибута="content" тип="java.lang.String">
        <description>более полное описание новости</description> <persistence
            type="property"
        />
    </атрибут>
</атрибуты>
</тип_элемента>
```

Интерактивный ярлык SAP Commerce 123: если вам не удается создать Новости itemtype или вы хотите пропустить этот шаг, замените <%HYBRIS\_HOME\_DIR%>/hybris/bin/custom/concerttours/resources/concerttours-items.xml c содержанием <%HYBRIS\_HOME\_DIR%>/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-itemsWithNews.xml

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-itemsWithNews.xml $HYBRIS_HOME_DIR/hybris/bin/custo
```

копировать /у %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\resources\concerttours-itemsWithNews.xml %HYBRIS\_HOME\_DIR%\hybris\bi

. Создайте класс слушателя, который прослушивает новые события Band висточником концертные туры расширение.

```
упаковка концертные туры.мероприятия;
импорт de.hybris.platform.servicelayer.event.events.AfterItemCreationEvent;
импорт de.hybris.platform.servicelayer.event.impl.AbstractEventListener;
импорт de.hybris.platform.servicelayer.model.ModelService;
импорт java.util.Date;
импорт концертные туры.модель.BandModel;
импорт концертные туры.модель.NewsModel;

открытый класс NewBandEventListener расширяет AbstractEventListener<AfterItemCreationEvent> {

    private static final String NEW_BAND_HEADLINE = "Новая группа, %s";
    private static final String NEW_BAND_CONTENT = "В городе появилась новая группа под названием %s. Скоро будут объявлены новости о type:"; private
        МодельСервис модельСервис;
    публичный МодельСервис получитьModelService()
    {
        возвращаться модельСервис;
    }
    public void setModelService(final ModelService modelService) {

        этот.modelService = modelService;
    }
    @Переопределить
    защищенный void onEvent(финальное событие AfterItemCreationEvent) {

        если (событие != null && событие.getSource() != null) {

            конечный объект Object = modelService.get(event.getSource()); если (объект
            instanceof BandModel)
            {
                окончательная BandModel полоса = (BandModel) объект;
                окончательный заголовок строки = String.format(NEW_BAND_HEADLINE, band.getName());
                окончательный контент строки = String.format(NEW_BAND_CONTENT, band.getName());
                окончательный NewsModel новости = modelService.create(NewsModel.class); news.setDate(new
                Date());
                news.setHeadline(заголовок);
                news.setContent(контент);
                modelService.save(новости);
            }
        }
    }
}
```

Интерактивный ярлык SAP Commerce 123: если вам не удается создать NewBandEventListener.java класс слушателя или вы хотите пропустить этот шаг, замените <%HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/events/NewBandEventListener.javac содержанием <%HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/events/NewBandEventListener.java

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/events/NewBandEventListener.java \$HYBRIS\_HOME\_DIR/hybris

echo f | xcopy %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\events\NewBandEventListener.java %HYBRIS\_HOME\_DIR%

. Зарегистрируйте нового слушателя в Spring, добавив конфигурацию Событие Слушатель Весенняя фасоль определение концертные туры-весна.xml в ресурсах папка концертные туры расширение.

```
<bean id="concerttourEventListener" class="concerttours.events.NewBandEventListener" parent="abstractEventListener">
    <имя_свойства="modelService" ref="modelService" /> </bean>
```

Интерактивный ярлык SAP Commerce 123: если вам не удается добавить конфигурацию Событие Слушатель Spring bean или вы хотите пропустить этот шаг, замените <%HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttours-spring.xmlc содержанием <%HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/events/NewBandEventListener.java

cp \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-spring-withListener.xml \$HYBRIS\_HOME\_DIR/hybris/bin

копировать /у %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\resources\concerttours-spring-withListener.xml %HYBRIS\_HOME\_DIR%\hybris

. Перестройте SAP Commerce с помощью Ant.

cd \$HYBRIS\_HOME\_DIR/hybris/bin/platform; ant очистить все

cd %HYBRIS\_HOME\_DIR%\hybris\bin\platform и очистите все

. Запустите SAP Commerce.

cd \$HYBRIS\_HOME\_DIR/hybris/bin/platform; ./hybrisserver.sh start

cd %HYBRIS\_HOME\_DIR%\hybris\bin\platform и запустить hybrisserver.bat

. Повторная инициализация в консоли администрирования SAP Commerce (консоль администрирования).

а. Войдите в SAP Commerce Administration Console (Administration Console) по адресу https://localhost:9002 используя логин, админ и пароль, который вы определили как переменную среды.

б. Перейдите в Инициализация платформы или перейдите прямо https://localhost:9002/platform/init.

в. В Настройки данных проекта убедитесь, что все флагги установлены.

г. Нажмите кнопку Инициализировать кнопку.

**Project data settings**

[Toggle all](#)

- [core](#)
- [scripting](#)
- [mediaweb](#)
- [commons](#)
- [processing](#)
- [impex](#)
- [validation](#)
- [catalog](#)
- [europe1](#)
- [platformservices](#)
- [workflow](#)
- [oauth2](#)
- [hac](#)
- [comments](#)
- [advancedsavedquery](#)
- [yhacext](#)
- [yempty](#)
- [concerttours](#)

**Patches**

[Initialize](#)

. Проверьте, что новости были успешно созданы.

а. В консоли администрирования SAP Commerce перейдите в Консоль вкладку и выберите ГибкийПоисквариант.

Теги ГибкийПоиск появляются страница.

б) Извлеките все новости из базы данных, введя запросы выберите {pk}, {date}, {headline}, {content} из {News} в ГибкийПоиск текстовое поле запроса и щелкните Выполнить кнопка.

. Запустите тест НовостиСобытия повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testNewsEvents test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testNewsEvents test
```

. Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add .; git commit -m "Создать прослушиватель"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Создать прослушиватель"
```

## Перехватчики и пользовательские события

Перехватчики перехватывают переходы жизненного цикла объектов модели и, в зависимости от условий этого перехода, могут публиковать событие, когда они это делают.

Классы моделей представляют элементы SAP Commerce. Каждая модель содержит все атрибуты элементов из всех расширений, что унифицирует доступ к данным для элемента.

Перехватчик адресует определенный шаг в жизненном цикле модели. Когда жизненный цикл достигает определенного шага, вы можете активировать соответствующий перехватчик. Перехватчик может изменить модель, вызвать исключение для прерывания текущего шага или опубликовать событие, если модель соответствует определенным критериям. Например, вы можете проверить, что атрибут содержит определенные значения, прежде чем сохранять модель.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[События и слушатели](#)

Следующий:[События, учитываемые кластер](#)

Сопутствующая информация

[Модели](#)

[Перехватчики](#)

## Создать перехватчик

Создайте перехватчик, который выдает исключение, если кто-то пытается сохранить группу элементов с отрицательными продажами альбомов и публикует пользовательское событие, если продажи альбомов группы превышают определенное число.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.java.

```
public void testEventInterceptorIntegrationTest() {
    assertTrue(checkTestSuiteXMLMatches("(.*testsuite ошибки=""0"" неудачи=""0"" (.*) имя=""BandAlbumSalesEventListenerIntegrationTest"" пакет=""co")
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testEventInterceptorIntegrationTest test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testEventInterceptorIntegrationTest test
```

### Процедура

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Создайте новый класс событий.

```
упаковка концертные туры.мероприятия;
импорт de.hybris.platform.servicelayer.event.events.AbstractEvent;
открытый класс BandAlbumSalesEvent расширяет AbstractEvent {

    private final Код строки; private
    final Имя строки; private final
    Длинные продажи;
    public BandAlbumSalesEvent(конечный код строки, конечное имя строки, конечные длинные продажи) {

       スーパー();
        этот.код      = КОД;
        это.имя       = ИМЯ;
        это.продажи   = продажи;
    }
    публичная строка получитьКод()
    {
        Код возврата;
    }
    публичная строка получитьИмя()
    {
        возвращаться ИМЯ;
    }
    публичный Длинный получитьПродажи()
    {
        возвращаться продажи;
    }
    @Переопределить
    публичная строка toString()
    {
        вернуть это.имя;
    }
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать класс событий или хотите пропустить этот шаг, замените

```
<HYBRIS_HOME_DIR>/hybris/bin/custom/concerttours/src/concerttours/events/BandAlbumSalesEvent.javac содержанием <HYBRIS_HOME_DIR>/hybris123/
src/main/webapp/resources/concerttours/src/concerttours/events/BandAlbumSalesEvent.java
```

```
то же самое $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/events/BandAlbumSalesEvent.java $HYBRIS_HOME_DIR/hybris
```

```
echo f | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\events\BandAlbumSalesEvent.java %HYBRIS_HOME
```

. Создайте перехватчик для пользовательского события.

```
упаковка концертные туры.перехватчики;
импорт статический de.hybris.platform.servicelayer.model.ModelContextUtils.getItemModelContext;
импорт
импорт de.hybris.platform.servicelayer.interceptor.InterceptorContext;
импорт de.hybris.platform.servicelayer.interceptor.InterceptorException;
импорт de.hybris.platform.servicelayer.interceptor.PrepareInterceptor;
импорт de.hybris.platform.servicelayer.interceptor.ValidateInterceptor;
импорт org.springframework.beans.factory.annotation.Autowired;
импорт concerttours.events.BandAlbumSalesEvent;
импорт концертные туры.модель.BandModel;

открытый класс BandAlbumSalesInterceptor реализует ValidateInterceptor, PrepareInterceptor {

    частный статический финальный длинный BIG_SALES = 50000L;
    частный статический финальный длинный NEGATIVE_SALES = 0L;
    @Autowired
    частный EventService eventService; @Override

    public void onValidate(final Object model, final InterceptorContext ctx) выдает InterceptorException {

        если (модель экземпляра BandModel) {
```

```

окончательная группа BandModel = (BandModel) модель;
окончательные длинные продажи =band.getAlbumSales();
если (продажи != null && продажи.longValue() < NEGATIVE_SALES) {
    throw new InterceptorException("Продажи альбомов должны быть положительными");
}
}
@Переопределить
public void onPrepare(final Object model, final InterceptorContext ctx) выдает InterceptorException {
    если (модель экземпляра BandModel) {
        окончательная BandModel полоса = (BandModel)
        модель; если (hasBecomeBig(band, ctx))
        {
            eventService.publishEvent(new BandAlbumSalesEvent(band.getCode(), band.getName(), band.getAlbumSales()));
        }
    }
private boolean hasBecomeBig(final BandModel band, final InterceptorContext ctx) {
    окончательные длинные продажи =band.getAlbumSales();
    если (продажи != null && продажи.longValue() >= BIG_SALES) {
        если (ctx.isNew(band))
        {
            вернуть истину;
        }
        еще
        {
            final Long oldValue = getItemModelContext(band).getOriginalValue(BandModel.ALBUMSALES); если (oldValue == null || oldValue.intValue() < BIG_SALES)
            {
                вернуть истину;
            }
        }
    }
    вернуть ложь;
}
}

```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать перехватчик или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/interceptors/BandAlbumSalesInterceptor.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/interceptors/BandAlbumSalesInterceptor.java

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/interceptors/BandAlbumSalesInterceptor.java \$HYBRIS\_HOME\_DIR

```
echo f | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\interceptors\BandAlbumSalesInterceptor.java
```

. Создайте прослушиватель событий.

```

упаковка концертные туры.мероприятия;
импорт de.hybris.platform.servicelayer.event.impl.AbstractEventListener;
импорт de.hybris.platform.servicelayer.model.ModelService;
импорт java.util.Date;
импорт концертные туры.модель.НовостиМодель;
```

открытый класс BandAlbumSalesEventListener расширяет AbstractEventListener<BandAlbumSalesEvent> {

```

private static final String BAND_SALES_HEADLINE = "%s продажи альбома превысили 50000"; private static final
String BAND_SALES_CONTENT = "%s продажи альбома указаны как %d"; private
    МодельСервис модельСервис;
публичный МодельСервис получитьModelService()
{
    возвращаться модельСервис;
}
public void setModelService(final ModelService modelService) {

    этот.modelService = modelService;
}
@Переопределить
защищенный void onEvent(финальное событие BandAlbumSalesEvent) {

    если (событие != null) {

        окончательный заголовок строки = String.format(BAND_SALES_HEADLINE, event.getName());
        окончательный String content = String.format(BAND_SALES_CONTENT, event.getName(), event.getSales()); окончательный
        NewsModel news = modelService.create(NewsModel.class);
        news.setDate(новая дата());
        news.setHeadline(заголовок);
        news.setContent(контент);
        modelService.save(новости);
    }
}
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать прослушиватель событий или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/events/BandAlbumSalesEventListener.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/events/BandAlbumSalesEventListener.java

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/events/BandAlbumSalesEventListener.java \$HYBRIS\_HOME\_DIR

```
echo f | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\events\BandAlbumSalesEventListener.java %HYBRIS_HOME_DIR
```

. Зарегистрируйте перехватчик и прослушиватель событий.

```

<bean id="bandAlbumSalesInterceptor" class="concerttours.interceptors.BandAlbumSalesInterceptor" />
<bean id="BandInterceptorMapping" class="de.hybris.platform.servicelayer.interceptor.impl.InterceptorMapping">
    <имя_свойства="interceptor" ref="bandAlbumSalesInterceptor" />
    <имя_свойства="typeCode" value="Band" />
</bean>
<bean id="bandAlbumSalesEventListener" class="concerttours.events.BandAlbumSalesEventListener" parent="abstractEventListener" >
```

```
<имя_свойства="modelService" ref="modelService" /> </bean>
```

Интерактивный ярлык SAP Commerce 123: если вам не удается зарегистрировать перехватчик и прослушиватель событий или вы хотите пропустить этот шаг, замените <*HYBRIS\_HOME\_DIR*>/hybris/bin/custom/concerttours/resources/concerttours-spring.xmlc содержанием <*HYBRIS\_HOME\_DIR*>/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-spring-withInterceptor.xml

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-spring-withInterceptor.xml $HYBRIS_HOME_DIR/hybris
```

```
копировать /у %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\resources\concerttours-spring-withInterceptor.xml %HYBRIS_HOME_DIR%\
```

.Напишите интеграционный тест, создав новый тестовый класс BandAlbumSalesEventListenerIntegrationTest, подтесты гспапка концертные туры расширение.

```
упаковка концертные туры.мероприятия;
импорт de.hybris.bootstrap.annotations.IntegrationTest;
импорт de.hybris.platform.servicelayer.ServicelayerTest;
импорт de.hybris.platform.servicelayer.exceptions.ModelSavingException;
импорт de.hybris.platform.servicelayer.model.ModelService;
импорт de.hybris.platform.servicelayer.search.FlexibleSearchService;
импорт java.lang.InterruptedException;
импорт java.util.List;
импорт java.util.concurrent.TimeUnit;
импорт javax.annotation.Resource;
импорт org.junit.Assert;
импорт org.junit.Do;
импорт org.junit.Test;
импорт concerttours.model.BandModel;
импорт concerttours.model.NewsModel;
импорт de.hybris.platform.core.Registry;
импорт org.springframework.jdbc.core.JdbcTemplate;

@Интеграционный тест
открытый класс BandAlbumSalesEventListenerIntegrationTest расширяет ServicelayerTest {

    @Ресурс
    частный ГибкийПоисковыйСервис гибкийПоисковыйСервис;
    @Resource
    частный ModelService modelService;
    частная статическая финальная строка BAND_CODE = "101-JAZ"; частная
    статическая финальная строка BAND_NAME = "Tight Notes";
    private static final String BAND_HISTORY = "Новый современный джазовый коллектив из 7 человек из Лондона, образованный в 2015 году";
    private static final Long MANY_ALBUMS_SOLD = Long.valueOf(1000000L);
    @До
    public void setUp() выдает исключение {

        пытаться {
            Thread.sleep(TimeUnit.SECONDS.toMillis(1));
            new JdbcTemplate(Registry.getCurrentTenant().getDataSource()).execute("CHECKPOINT");
            Thread.sleep(TimeUnit.SECONDS.toMillis(1));
        }
        catch (InterruptedException exc) {}
        createCoreData();
        createDefaultCatalog();
    }
    @Test(ожидаемый = ModelSavingException.class) public void
    testValidationInterceptor()
    {
        окончательная BandModel band = modelService.create(BandModel.class);
        band.setCode(BAND_CODE);
        band.setAlbumSales(Long.valueOf(-10L));
        modelService.save(band);
    }

    @Тест
    public void testEventSending() выдает исключение {

        окончательная BandModel band = modelService.create(BandModel.class);
        band.setCode(BAND_CODE);
        band.setName(ИМЯ_ГРУППЫ);
        band.setHistory(ИСТОРИЯ_ГРУППЫ);
        band.setAlbumSales(ПРОДАННЫЕ_АЛЬБОМЫ_MANY);
        modelService.save(группа);
        окончательная модель новостей news = findLastNews();
        Assert.assertTrue("Неожиданные новости: " + news.getHeadline(), news.getHeadline().contains(BAND_NAME)); }

        частная NewsModel findLastNews()

        final StringBuilder builder = new StringBuilder(); builder.append("SELECT {n:").append(NewsModel.PK).append("} ");
        builder.append("FROM {").append(NewsModel._TYPECODE).append("} AS n"); builder.append(" ORDER BY ");
        builder.append("{n:").append(NewsModel.CREATIONTIME).append("} DESC");

        окончательный список<NewsModel> список = FlexibleSearchService.<NewsModel> поиск(builder.toString()).getResult(); если (list.isEmpty())

        {
            возвращаться нулевой;
        }
        еще
        {
            возвращаться список.получить(0);
        }
    }
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете написать интеграционный тест или хотите пропустить этот шаг, замените

```
<HYBRIS_HOME_DIR>/hybris/bin/custom/concerttours/testsrc/concerttours/events/BandAlbumSalesEventListenerIntegrationTest.javac содержанием
```

```
<HYBRIS_HOME_DIR>/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/events/BandAlbumSalesEventListenerIntegrationTe
```

```
то же самое $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/events/BandAlbumSalesEventListenerIntegrationTest.
```

```
echo f | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\testsrc\concerttours\events\BandAlbumSalesEventListenerIntegr
```

.Перестройте SAP Commerce с помощью Ant.

12/3/24, 11:01 утра

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ant очистить все
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и очистите все
```

Инициализируйте своего тестового арендатора.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; муравей юнитинит
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и ant yunitinit
```

. Запустите Группа Альбом Продажи Событие Слушатель Интеграция Тест проверьте и подтвердите, что вы видите найдено 7 тест классов в выводе журнала.

```
муравей интеграционные тесты - Dtestclasses.packages="concerttours.*"
```

```
муравей интеграционные тесты - Dtestclasses.packages=концертные туры.*
```

. Посмотреть результаты теста можно на <*%HYBRIS\_HOME\_DIR%*>/hybris/log/junit/index.html подтверждаю, что Группа Альбом Продажи Событие Слушатель Интеграция Тест тест пройден.

. Запустите testEventInterceptorIntegrationTest повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testEventInterceptorIntegrationTest test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testEventInterceptorIntegrationTest тест
```

. Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add . ; git commit -m "Создать пользовательское событие"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Создать пользовательское событие"
```

## События, учитывающие кластер

SAP Commerce поддерживает события, поддерживающие кластер. Благодаря событиям, поддерживающим кластер, SAP Commerce может обрабатывать события в отдельных потоках или на определенных узлах кластера.

По умолчанию SAP Commerce обрабатывает все события синхронно. Но синхронный обмен сообщениями имеет некоторые недостатки:

- Основной поток ждет, пока все события не будут обработаны. Например, группа не сохраняется, пока не будет создана соответствующая новость.
- Медленный прослушиватель может привести к тому, что система выдаст исключение тайм-аута.

Чтобы избежать таких недостатков, в некоторых обстоятельствах может быть полезно обрабатывать события асинхронно. Чтобы использовать этот тип обработки событий, рассмотрите возможность использования событий, поддерживающих кластер, особенно если вы хотите, чтобы ваши события достигали выбранных или всех узлов в экземпляре SAP Commerce, развернутом в кластере в производственной среде.

-Примечание

Невозможно предоставить полную гарантию того, что данное событие всегда обрабатывается на выбранном узле, поскольку временные сбои могут привести к тому, что события не будут обработаны правильно, что сделает их менее надежными для критически важных для бизнеса действий. Для более надежной асинхронной обработки используйте службу задач, чья логика повтора и управление состоянием обеспечивают лучшую устойчивость к сбоям. Для получения дополнительной информации о службе задач см. [Служба задач](#).

Информацию о потенциальных рисках, связанных с использованием событий, поддерживающих кластеры, см. [События, учитывающие кластер](#).

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Перехватчики и пользовательские события](#)

Следующий:[Задания Cron](#)

## Настройка асинхронного обмена сообщениями

Реализуйте асинхронный обмен сообщениями и обработку с учетом кластера, чтобы избежать потенциальных проблем со временем ожидания.

## Предпосылки

В ваш комплект документов включен следующий приемочный тест Hybris123Tests.java.

```
public void testAsyncEventInterceptorIntegrationTest() { assertTrue(  
    checkTestSuiteXMLMatches("(.)testsuite ошибки=\"0\" неудачи=\"0\" (.*) имя=\"BandAlbumSalesEventListenerIntegrationTest\" пакет=\"concerttours.e.checkTestSuiteXMLMatches(.*)testcase  
    classname=\"concerttours.events.BandAlbumSalesEventListenerIntegrationTest\" имя=\"testEventSendingAsync\"(.*)")  
);
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testAsyncEventInterceptorIntegrationTest тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testAsyncEventInterceptorIntegrationTest тест
```

## Процедура

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Измените Группа Альбом Продажи События чтобы стать кластерно-ориентированным.

Добавляя реализует ClusterAwareEvent, затем определите новый метод публикации, который гарантирует, что события публикуются и обрабатываются только тем узлом, с которого они поступили.

```
упаковка концертные туры.мероприятия;
импорт de.hybris.platform.servicelayer.event.ClusterAwareEvent;
импорт de.hybris.platform.servicelayer.event.events.AbstractEvent;

открытый класс BandAlbumSalesEvent расширяет AbstractEvent реализует ClusterAwareEvent {

    private final Код строки; private final
    final Имя строки; private final
    Длинные продажи;
    public BandAlbumSalesEvent(конечный код строки, конечное имя строки, конечные длинные продажи) {

        super();
        этот.код      = Код;
        это.имя       = Имя;
        это.продажи   = продажи;
    }
    публичная строка получитьКод()
    {
        Код возврата;
    }
    публичная строка получитьИмя()
    {
        возвращаться Имя;
    }
    публичный Длинный получитьПродажи()
    {
        возвращаться продажи;
    }
    @Переопределить
    публичная строка toString()
    {
        вернуть это.имя;
    }
    @Переопределить
    public boolean publish(final int sourceNodeId, final int targetNodeId) {

        возврат (sourceNodeId == targetNodeId);
    }
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать кластерное событие самостоятельно или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/events/BandAlbumSalesEvent.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/events/BandAlbumSalesEventAsync.java

```
то же самое $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/events/BandAlbumSalesEventAsync.java $HYBRIS_HOME_DIR/
```

```
echo a | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\events\BandAlbumSalesEventAsync.java %HYBRIS
```

. Добавьте новый тест с именем testEventSendingAsync чтобы учесть возможную задержку и закомментировать @Тестаннотация для отключения testEventSending().

```
@Test
public void testEventSendingAsync() выдает исключение {

окончательная BandModel band = modelService.create(BandModel.class);
band.setCode(BAND_CODE);
band.setName(Имя_ГРУППЫ);
band.setHistory(ИСТОРИЯ_ГРУППЫ);
band.setAlbumSales(ПРОДАННЫЕ_АЛЬБОМЫ_MANY);
modelService.save(группа);
// добавьте здесьсон, чтобы дождаться завершения потока обработки событий
Thread.sleep(2000L);
окончательная модель новостей news = findLastNews();
Assert.assertTrue("Неожиданные новости: " + news.getHeadline(), news.getHeadline().contains(BAND_NAME)); }
```

Интерактивный ярлык SAP Commerce 123: если вы не можете обновить метод теста самостоятельно или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/testsrc/concerttours/events/BandAlbumSalesEventListenerIntegrationTest.javac содержанием

```
<HYBRIS_HOME_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/events/BandAlbumSalesEventListenerIntegrationTestAs
```

```
то же самое $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/events/BandAlbumSalesEventListenerIntegrationTestA
```

```
echo a | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\testsrc\concerttours\events\BandAlbumSalesEventListenerIntegrationIntegr
```

-Примечание

Тест расширяет ServicelayerTest, но не ServicelayerTransactionalTest. Это сделано намеренно. Если вы запустите этот тест в транзакции, вы можете столкнуться с проблемами конфигурации изоляции базы данных. Например, если вы запустите этот тест в транзакции, а уровень изоляции базы данных установлен повторяемое чтение, новость будет вне вашей тестовой транзакции, так как вы создаете новости в отдельном потоке. Другими словами, все, что фиксируется вне транзакции, не будет видно внутри транзакции (повторяемая изоляция чтения), что приведет к сбою теста. Изменение уровня изоляции на менее строгий (читать совершено) приведет к успешному прохождению теста.

. Перестройте SAP Commerce с помощью Ant.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ant очистить все
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и очистите все
```

. Инициализируйте своего тестового арендатора.

12/3/24, 11:01 утра

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; муравей юнитинит
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и ant yunitinit
```

. Запустите группу Альбом Продажи Событие Слушатель Интеграция Тест проверьте и подтвердите, что вы видите найдено 7 тест классов в выводе журнала.

```
муравей интеграционные тесты - Dtestclasses.packages="concerttours.*"
```

```
муравей интеграционные тесты - Dtestclasses.packages=концертные туры.*
```

. Посмотреть результаты теста можно на <*HYBRIS\_HOME\_DIR*>/hybris/log/junit/index.html подтверждают, что группа Альбом Продажи Событие Слушатель Интеграция Тест тест пройден.

. Запустите testAsyncEventInterceptorIntegrationTest повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testAsyncEventInterceptorIntegrationTest тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testAsyncEventInterceptorIntegrationTest тест
```

. Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add . ; git commit -m "Настройка асинхронного обмена сообщениями"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Настройка асинхронного обмена сообщениями"
```

## Задания Cron

SAP Commerce предоставляет средства для настройки регулярных задач. С помощью этих задач, или заданий cron, вы можете многократно выполнять сложную бизнес-логику в определенное время и с определенными интервалами.

Например, вы можете проводить инвентаризацию каждого воскресенья в полночь или уведомлять веб-администратора о пиковых нагрузках на серверах каждый час. Этого можно добиться с помощью комбинации выделенных классов для бизнес-логики и встроенной функциональности управления заданиями cron SAP Commerce.

Первым шагом в реализации задания cron является размещение вашей бизнес-логики в классе, который расширяет AnnotationWork выполнима. Затем вы разрабатываете новый, простой сервис для предоставления доступа к новостям и инкапсулируете эту бизнес-логику в класс заданий. Шаги аналогичны тем, которые вы предприняли для создания сервиса группы, и служат для укрепления этого шаблона.

Платформа SAP Commerce поставляется с полезным классом utilit электронной почты, MailUtils, что упрощает отправку писем с помощью API общей почты. Вы используете MailUtils класс для создания объекта сообщения электронной почты, заполненного значениями из местных свойств в вашем конфигурации каталога.

Родительская тема: [Изучите тип SAP Commerce](#)

Предыдущий: [События, учитываемые кластером](#)

Следующий: [Триггеры](#)

Сопутствующая информация

[Служба Cronjobs](#)

## Напишите бизнес-логику для задания Cron

Напишите задание cron для отправки ежедневных сводок новостей.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест Hybris123Tests.java.

```
public void testSendNewsJobIntegrationTest() {  
    assertTrue(checkTestSuiteXMLMatches("(.* )testsuite ошибки=\"0\" неудачи=\"0\" (.*) имя=\"SendNewsJobIntegrationTest\" пакет=\"concerttours.jobs\"})
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testSendNewsJobIntegrationTest тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testSendNewsJobIntegrationTest тест
```

### Процедура

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Создайте интерфейс доступа к новостным данным в <*HYBRIS\_HOME\_DIR*>/hybris/bin/custom/concerttours/src/concerttours/daos/NewsDAO.java

```
упаковка концертные туры.daos;  
импорт java.util.Date;  
импорт java.util.List;  
импорт концертные туры.Модель.НовостиМодель;
```

```
публичный интерфейс NewsDAO {
    Список<NewsModel> getNewsOfTheDay(Дата дата);
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать интерфейс доступа к новостным данным самостоятельно или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/daos/NewsDAO.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/daos/NewsDAO.java

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/daos/NewsDAO.java \$HYBRIS\_HOME\_DIR/hybris/bin/custom/c

```
echo f | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\daos\NewsDAO.java %HYBRIS_HOME_DIR%\hybris\b
```

. Создайте реализацию доступа к новостным данным в <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/daos/impl/DefaultNewsDAO.java

пакет concerttours.daos.impl;

```
импорт de.hybris.platform.servicelayer.search.FlexibleSearchQuery;
импорт de.hybris.platform.servicelayer.search.FlexibleSearchService;
импорт java.text.SimpleDateFormat;
импорт java.util.Календарь;
импорт java.util.Коллекции;
импорт java.util.Дата;
импорт java.util.List;
импорт org.springframework.beans.factory.annotation.Autowired;
импорт org.springframework.stereotype.Component;
импорт concerttours.daos.NewsDAO;
импорт concerttours.model.NewsModel;
```

```
@Component(значение = "newsDAO")
публичный класс DefaultNewsDAO реализует NewsDAO {
```

```
частная статическая окончательная строка SQL_DATE_FORMAT = "yyyy-MM-dd";
@Autowired
частный ГибкийПоисковыйСервис гибкийПоисковыйСервис;
@Override
public List<NewsModel> getNewsOfTheDay(final Date date) {
    если (дата == null) {
        вернуть Коллекции.emptyList();
    }
    окончательная строка theDay = new SimpleDateFormat(SQL_DATE_FORMAT).format(date);
    окончательная строка theNextDay = new SimpleDateFormat(SQL_DATE_FORMAT).format(oneDayAfter(date));
    окончательная строка queryString = //
        "ВЫБРАТЬ (p." + NewsModel.PK + ") //"
        + "FROM (" + NewsModel._TYPECODE + " AS p) //"
        + "ГДЕ {date} >= ДАТА '\"" + theDay + "\"' //"
        + "И {дата} <= ДАТА '\"" + theNextDay + "\"'";
    окончательный запрос FlexibleSearchQuery = new FlexibleSearchQuery(queryString); return
    FlexibleSearchService.<NewsModel> search(query).getResults();
}
частная Дата oneDayAfter(финальная Дата date) {
    окончательный Календарь cal = Calendar.getInstance();
    cal.setTime(date);
    cal.add(Calendar.DATE, 1); return
    cal.getTime();
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать реализацию самостоятельно или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/daos/impl/DefaultNewsDAO.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/daos/impl/DefaultNewsDAO.java

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/daos/impl/DefaultNewsDAO.java \$HYBRIS\_HOME\_DIR/hybris/

```
echo f | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\daos\impl\DefaultNewsDAO.java %HYBRIS_HOME_D
```

. Создайте интерфейс новостной службы под названием NewsService.java в <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/service

упаковка концертные туры.сервис;
импорт java.util.Дата;
импорт java.util.List;
импорт концертные туры.модель.НовостиМодель;

```
публичный интерфейс NewsService {
    Список<NewsModel> getNewsOfTheDay(Дата дата);
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать интерфейс службы новостей самостоятельно или хотите пропустить этот шаг, скопируйте <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/service/NewsService.java в <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/service

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/service/NewsService.java \$HYBRIS\_HOME\_DIR/hybris/bin/c

```
echo a | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\service\NewsService.java %HYBRIS_HOME_DIR%\h
```

. Создайте реализацию новостной службы под названием DefaultNewsService.java в <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/service/impl

```
упаковка concerttours.service.impl;
импорт java.util.Date;
импорт java.util.List;
импорт org.springframework.beans.factory.annotation.Autowired;
импорт concerttours.daos.NewsDAO;
импорт концертные туры.модель.НовостиМодель;
импорт концертные туры.сервис.НовостиСервис;
```

```
публичный класс DefaultNewsService реализует NewsService {
    @Autowired
    частные новостиDAO новостиDAO;

    public void setNewsDAO(final NewsDAO newsDAO) {
        this.newsDAO = newsDAO;
    }

    @Переопределить
    public List<NewsModel> getNewsOfTheDay(final Date date) {
        вернуть новостиDAO.getNewsOfTheDay(дата);
    }
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать реализацию самостоятельно или хотите пропустить этот шаг, скопируйте <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/service/impl/DefaultNewsService.java в <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/service/impl

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/service/impl/DefaultNewsService.java \$HYBRIS\_HOME\_DIR/

echo a | xcopy %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\service\impl\DefaultNewsService.java %HYBRIS

.Настройте сервер электронной почты, добавив свойства сервера электронной почты в `вместные.свойства`.

В примере показано, как это сделать с помощью общей учетной записи электронной почты. Обновите SMTP-сервер, имя пользователя и пароль, чтобы они совпадали с вашими.

а. Добавьте следующие свойства в `вместные.свойства` в каталоге `cong`, отредактируйте их, убедившись, что заменили `<smtp.email.com>`, `<ваше имя пользователя>`, `<ваш пароль>`, и `<ваш адрес электронной почты>` с вашим именем пользователя, паролем и адресом электронной почты. Если порт 465 недоступен, измените порт.

```
mail.from=concerttours-no-reply@hybris.de
mail.replyto=concerttours-no-reply@hybris.de
mail.smtp.server=smtp.email.com
mail.smtp.port=465

mail.smtp.пользователь=<ваше имя
пользователя> mail.smtp.пароль=<ваш пароль>
news_summary_mailing_address=<ваш
электронная почтаадрес>
```

6. Обновите свою учетную запись электронной почты, чтобы (временно) разрешить менее безопасным приложениям получать к ней доступ, выбрав `Разрешить менее безопасные приложения` вариант.

Интерактивный ярлык SAP Commerce 123: если вы не можете добавить свойства самостоятельно или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/config/local.properties с содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/resources/config/local.properties

cat \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/resources/config/local.properties >> \$HYBRIS\_HOME\_DIR/hybris/config/local

типа %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\resources\config\local.properties >> %HYBRIS\_HOME\_DIR%\hybris\config\lo

.Создайте класс работы.

```
упаковка концертные туры.работа;
импорт de.hybris.platform.cronjob.enums.CronJobResult;
импорт de.hybris.platform.cronjob.enums.CronJobStatus;
импорт de.hybris.platform.cronjob.model.CronJobModel;
импорт de.hybris.platform.servicelayer.config.ConfigurationService;
импорт de.hybris.platform.servicelayer.cronjob.AbstractJobPerformable;
импорт de.hybris.platform.servicelayer.cronjob.PerformResult;
импорт de.hybris.platform.util.mail.MailUtils;
импорт java.util.Дата;
импорт java.util.List;
импорт org.apache.commons.mail.Email;
импорт org.apache.commons.mail.EmailException;
импорт org.apache.commons.configuration.Configuration;
импорт org.apache.log4j.Logger;
импорт org.springframework.beans.factory.annotation.Required;
импорт concerttours.model.NewsModel;
импорт концертные туры.сервис.НовостиСервис;

открытый класс SendNewsJob расширяет AbstractJobPerformable<CronJobModel> {

    частный статический финальный Logger LOG = Logger.getLogger(SendNewsJob.class);
    частный НовостиСервис новостиСервис; КонфигурацияСервис;
    частный конфигурацияСервис;

    @Необходимый
    публичная служба новостей getNewsService()
    {
        возврат новостейСервис;
    }

    @Необходимый
    публичная служба конфигурации { getConfigService()

        возврат конфигурацииService;
    }

    @Необходимый
    public void setNewsService(final NewsService newsService)

        this.newsService = newsService;
    }

    @Необходимый
    public void setConfigService(final ConfigurationService configurationService)

        this.configurationService = configurationService;
    }

    @Переопределить
    public PerformResult выполнить(final CronJobModel cronJob)
    {

        LOG.info("Отправка новостных писем. Обратите внимание, что org.apache.commons.mail.send() может блокироваться, если находится за брандмауэром/прокси-сервером.");
    }
}
```

LOG.info("Отправка новостных писем. Обратите внимание, что org.apache.commons.mail.send() может блокироваться, если находится за брандмауэром/прокси-сервером.");

12/3/24, 11:01 утра

```
окончательный список<NewsModel> newsItems = getNewsService().getNewsOfTheDay(new Date()); если
(newsItems.isEmpty())
{
    LOG.info("Нет новостей на сегодня, пропускается отправка рейтинговых писем"); return new
    PerformResult(CronJobResult.SUCCESS, CronJobStatus.FINISHED);
}
окончательный StringBuilder mailContentBuilder = new StringBuilder(2000); int index = 1;

mailContentBuilder.append("Сводка сегодняшних новостей:\n\n"); для
(final NewsModel news : newsItems)
{
    mailContentBuilder.append(index++);
    mailContentBuilder.append(" " );
    mailContentBuilder.append(news.getHeadline());
    mailContentBuilder.append("\n");
    mailContentBuilder.append(news.getContent());
    mailContentBuilder.append("\n\n");
}

пытаюсь
{
    sendEmail(mailContentBuilder.toString());
}
поймать (финальное исключение EmailException e)

    LOG.error("Проблема с отправкой нового письма. Обратите внимание, что org.apache.commons.mail.send() может блокироваться, если находится за брандмауэром/прокси-сервером.");
    LOG.error("Проблема с отправкой нового письма.", e);
    вернуть новый PerformResult(CronJobResult.FAILURE, CronJobStatus.FINISHED);
}
вернуть новый PerformResult(CronJobResult.SUCCESS, CronJobStatus.FINISHED);
}

private void sendEmail(final String message) выдает EmailException {

final String subject = "Ежедневная сводка новостей"; //
получить конфигурацию почтовой службы
окончательный Email email = MailUtils.getPreConfiguredEmail(); //
отправить сообщение
Конфигурация config = getConfigurationService().getConfiguration(); Стока получателя =
config.getString("news_summary_mailing_address", null); email.addTo(recipient);

email.setSubject(тема);
email.setMsg(сообщение);
email.setTLS(истина);
электронная почта отправителя;
LOG.info(тема);
LOG.info(сообщение);
}
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать класс задания самостоятельно или хотите пропустить этот шаг, замените  
<HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/jobs/SendNewsJob.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/  
src/main/webapp/resources/concerttours/src/concerttours/jobs/SendNewsJob.java

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/jobs/SendNewsJob.java \$HYBRIS\_HOME\_DIR/hybris/bin/cust

эхо ж | xcopy %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\jobs\SendNewsJob.java %HYBRIS\_HOME\_DIR%\hybris

.Зарегистрируйте услугу и работу весной.

```
<псевдоним name="defaultNewsService" псевдоним="newsService" />
<bean id="defaultNewsService" class="concerttours.service.impl.DefaultNewsService">
    <имя_свойства="newsDAO" ref="newsDAO" /> </bean>

<bean id="sendNewsJob" class="concerttours.jobs.SendNewsJob" parent="abstractJobPerformable">
    <имя_свойства="newsService" ref="newsService" />
    <имя_свойства="configurationService" ref="configurationService" /> </bean>
```

Интерактивный ярлык SAP Commerce 123: если вы не можете зарегистрировать службу и задание самостоятельно или хотите пропустить этот шаг, замените  
<HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttours-spring.xmlc содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/  
resources/concerttours/resources/concerttours-spring-withNewsService.xml

cp \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-spring-withNewsService.xml \$HYBRIS\_HOME\_DIR/hybris/

копировать /у %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\resources\concerttours-spring-withNewsService.xml %HYBRIS\_HOME\_DIR%\

.Напишите интеграционный тест для проверки успешного создания рабочих мест.

```
упаковка концертные туры.работа;
импорт de.hybris.bootstrap.annotations.IntegrationTest;
импорт de.hybris.platform.cronjob.enums.CronJobResult;
импорт de.hybris.platform.servicelayer.servicelayer.TransactionTest;
импорт de.hybris.platform.servicelayer.cronjob.PerformResult;
импорт de.hybris.platform.servicelayer.model.ModelService; java.lang.InterruptedException;
импорт
импорт java.util.Date;
импорт java.util.concurrent.TimeUnit;
импорт de.hybris.platform.core.Registry;
импорт org.springframework.jdbc.core.JdbcTemplate;
импорт javax.annotation.Resource;
импорт org.junit.Uтверждение;
импорт org.junit.После;
импорт org.junit.До;
импорт org.junit.Test;
импорт концертные туры.модель.НовостиМодель;
```

@Интеграционный тест
открытый класс SendNewsJobIntegrationTest расширяет ServicelayerTransactionalTest {

```
@Ресурс
частная модельСервис     модельСервис;
@Ресурс
частный ОтправитьНовостиРабота отправитьНовостиРабота;
```

@До
public void setUp() выдает исключение

12/3/24, 11:01 утра

```
{  
    пытаться {  
        Thread.sleep(TimeUnit.SECONDS.toMillis(1));  
        new JdbcTemplate(Registry.getCurrentTenant().getDataSource()).execute("CHECKPOINT");  
        Thread.sleep(TimeUnit.SECONDS.toMillis(1));  
    }  
    поймать (InterruptedException exc) {}  
  
    @Test  
    публичный пустота тестNoNewsItems()  
{  
    окончательный результат PerformResult = sendNewsJob.perform(null);  
    Assert.assertEquals("Задание выполнено неправильно", CronJobResult.SUCCESS, result.getResult());  
}  
  
    @Test  
    public void testSendingNews() выдает исключение {  
  
    окончательная NewsModel news1 = modelService.create(NewsModel.class);  
    news1.setHeadline("test заголовок 1");  
    news1.setContent("текст содержание 1");  
    news1.setDate(новая дата());  
    окончательная NewsModel news2 = modelService.create(NewsModel.class);  
    news2.setHeadline("test заголовок 2");  
    news2.setContent("текст содержание 2");  
    news2.setDate(новая дата());  
  
    окончательный результат PerformResult = sendNewsJob.perform(null);  
    Assert.assertEquals("Задание выполнено неправильно", CronJobResult.SUCCESS, result.getResult());  
}  
  
    @После  
    public void tearDown() {  
    }  
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать интеграционный тест самостоятельно или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/testsrc/concerttours/jobs/SendNewsJobIntegrationTest.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/jobs/SendNewsJobIntegrationTest.java

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/jobs/SendNewsJobIntegrationTest.java \$HYBRIS\_HOME\_

echo f | xcopy %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\testsrc\concerttours\jobs\SendNewsJobIntegrationTest.java %HY

. Перестройте SAP Commerce с помощью Ant.

cd \$HYBRIS\_HOME\_DIR/hybris/bin/platform; ant очистить все

cd %HYBRIS\_HOME\_DIR%\hybris\bin\platform и очистите все

. Инициализируйте своего тестового арендатора.

cd \$HYBRIS\_HOME\_DIR/hybris/bin/platform; муравей юнитинит

cd %HYBRIS\_HOME\_DIR%\hybris\bin\platform и ant yunitinit

. Запустите интеграционный тест и убедитесь, что ОтправитьНовостиРаботаИнтеграцияТест проходит.

муравей интеграционные тесты - Dtestclasses.packages="concerttours.\*"

муравей интеграционные тесты - Dtestclasses.packages=концертные туры.\*

. Посмотреть результаты теста можно на <HYBRIS\_HOME\_DIR>/hybris/log/junit/index.html.

. Запустите testSendNewsJobIntegrationTest повторите приемочное испытание и подтвердите, что оно теперь пройдено.

cd \$HYBRIS\_HOME\_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testSendNewsJobIntegrationTest test

cd %HYBRIS\_HOME\_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testSendNewsJobIntegrationTest test

. Зафиксируйте изменения в локальном репозитории Git.

cd \$HYBRIS\_HOME\_DIR/hybris; git add . ; git commit -m "Написать бизнес-логику для задания Cron"

cd %HYBRIS\_HOME\_DIR%\hybris & git add . & git commit -m "Написать бизнес-логику для задания Cron"

## Триггеры

Успешно внедрив бизнес-логику в класс задания, вы можете запустить его выполнение с помощью задания cron.

Задание cron состоит изработакласс, содержащий бизнес-логику, и триггер для запуска задания через регулярные интервалы. Первый шаг в настройке нового задания cron — уведомить SAP Commerce о вашем новом класс, создавая ваши основные данные. Во время создания основных данных, ServicelayerРаботаэлемент создается для каждого определения Spring, которое имеет класс, реализующий JobPerformable Интерфейс. Атрибут кода каждого нового элемента задания устанавливается на имя соответствующего компонента Spring.

Один раз вServicelayerРаботаэлемент создан, вы можете создать задание cron, чтобы обернуть егоServicelayerРабота, и определить триггер, который его запускает.

Выражение cron представляет собой строку, состоящую из 6 или 7 полей, разделенных пробелом. Поля могут содержать любые допустимые значения, а также различные комбинации допустимых специальных символов для этого поля.

Имя поля	Обязательный	Допустимые значения	Разрешенные специальные символы
Секунды	ДА	0-59	, - * /
Минуты	ДА	0-59	, - * /
Часы	ДА	0-23	, - * /
День месяца	ДА	31-янв.	, - * ? / ДВ
Месяц	ДА	1-12 или ЯНВ-ДЕК	, - * /
День недели	ДА	1-7 или ВС-СБ	, - * ? / Л #
Год	НЕТ	пустой, 1970-2099	, - * /

Первое задание сценария, которое вы создаете для своего расширения, запускается ежедневно и отправляет сводки новых элементов по электронной почте на указанный адрес электронной почты или в список рассылки. Затем вы создаете второе задание сценария, используя язык сценариев вместо класса Java. Используя язык сценариев, вы можете добавлять задания сценария в систему без необходимости ее перестройки и повторного развертывания.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Задания Cron](#)

Следующий:[Кратко](#)

#### Сопутствующая информация

[Hybris пособие по триггеру Cron](#)

## Настройте задание Cron с помощью триггера с помощью Impex

Настройте задание сценария на запуск через регулярные промежутки времени.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.java.

```
public void testSendingNewsMails() выдает исключение {
долгое времяSinceLastMailWasSentMS = LogHelper.getMSSinceLastNewsMailsLogged();
assertTrue("Журнал последней отправленной почты должен был быть недавно помечен временем " + timeSinceLastMailWasSentMS,
timeSinceLastMailWasSentMS < 5*60*1000);
}
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testОтправкаНовостейПисьма тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testОтправкаНовостейПисьма тест
```

### Процедура

. Запустите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh start
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и запустить hybrisserver.bat
```

. Повторно инициализируйте консоль администрирования SAP Commerce (консоль администрирования).

а. Войдите в SAP Commerce Administration Console (Administration Console) по адресу <https://localhost:9002> используя логин, **админи** пароль, который вы определили как переменную среды.

б. Перейдите в **Инициализация платформы** или перейдите прямо <https://localhost:9002/platform/init>

в. В **Настройки данных проекта** убедитесь, что все флагги установлены.

г. Нажмите кнопку **Инициализировать**.

**Project data settings**

[Toggle all](#)

- [core](#)
- [scripting](#)
- [mediaweb](#)
- [commons](#)
- [processing](#)
- [impex](#)
- [validation](#)
- [catalog](#)
- [europe1](#)
- [platformservices](#)
- [workflow](#)
- [oauth2](#)
- [hac](#)
- [comments](#)
- [advancesavedquery](#)
- [yhacext](#)
- [yempty](#)
- [concerttours](#)

**Patches**

[Initialize](#)

Интерактивный ярлык SAP Commerce 123: если вы не можете инициализировать систему самостоятельно или хотите пропустить этот шаг, выполните следующую команду maven.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#simulateInitialization test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#simulateInitialization test
```

. Убедитесь, что ServicelayerJob был успешно создан.

а. В консоли администрирования SAP Commerce перейдите в Консоль ГибкийПоисквариант.

The ГибкийПоиск появляется страница.

б) Чтобы проверить, что новый элемент ServicelayerJob был создан, введите запросы выберите {код} из {servicelayerjob}, где {код} = 'sendNewsJob' в ГибкийПоиск текстовое поле запроса и щелкните Выполнить кнопка.

. Создайте задание cron и триггер.

а. В консоли администрирования SAP Commerce перейдите в Консоль вкладку и выберите Импекс Импортвариант.

б. Разверните область настроек, выберите Включить выполнение кода установите флагок и импортируйте скрипт impex.

```
INSERT_UPDATE
ServicelayerJob; код[unique=истина]; SpringId; ; отправитьНовостиЗадания; отправитьНовостиЗадания

# Определите задание cron и задание, которое оно включает
INSERT_UPDATE CronJob;
code[unique=true]; job(code); singleExecutable; sessionLanguage(isocode); sendNewsCronJob; sendNewsJob; false; de

# Определите триггер, который периодически вызывает задание cron
INSERT_UPDATE Trigger; cronJob(code)[unique=true]; cronExpression
# % afterEach: impex.getLastImportedItem().setActivationTime(new Date()); ; sendNewsCronJob; 0/10
* * * ?
```

Выход консоли показывает, что задание запускается каждые 10 секунд. В производственной среде вы бы изменили триггер на запуск один раз в конце дня.

Интерактивный ярлык SAP Commerce 123: если вы не можете запустить задание cron самостоятельно или хотите пропустить этот шаг, выполните следующую команду.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#simulateLoadingJobImpex test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#simulateLoadingJobImpex test
```

. Чтобы скрипт ImpEx загружался СоС, создайте essentialdataJobs.impex и внесите в него вспомогательные данные с тем же содержимым impex, чтобы эти элементы воссоздавались во время основной фазы загрузки данных любой операции инициализации или обновления.

Интерактивный ярлык SAP Commerce 123: если вам не удается создать essentialdataJobs.impex или вы хотите пропустить этот шаг, скопируйте <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/resources/script/essentialdataJobs.impexx <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/impex

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/resources/script/essentialdataJobs.impex \$HYBRIS\_HOME\_DIR/hybris/bin/cu

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Запустите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh start
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и запустить hybrisserver.bat
```

. Запустите тест Отправка Новостей Письма повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testОтправкаНовостейПисьма тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testОтправкаНовостейПисьма тест
```

. Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add . ; git commit -m "Настройте задание Cron с помощью триггера с помощью Impex"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Настройте задание Cron с помощью триггера с помощью Impex"
```

## Круто

Для написания заданий для выполнения можно использовать язык сценариев Groovy.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Триггеры](#)

Следующий:[Панель управления администрированием Back-ofice](#)

### Создание скрипта Cronjob с использованием Groovy

Напишите задание с использованием Groovy для изменения статуса утверждения концертов.

#### Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.java нале.

```
public void testGroovyScript() выдает исключение {
    canLoginToHybrisCommerce(); navigationTo("https://localhost:9002/console/
flexsearch");
    waitForFlexQueryFieldThenSubmit("SELECT {p.pk}, {p.code}, {p.name}, {q.code} FROM {Concert AS p}, {ArticleApprovalStatus AS q} WHERE {p.appro assertTrue(waitForText("check"));
}
```

}

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testGroovyScript тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testGroovyScript тест
```

#### Процедура

. Получить список концертов с указанием их статуса одобрения.

а. Войдите в SAP Commerce Administration Console (Administration Console) по адресу <https://localhost:9002> используя логин, админ пароль, который вы определили как переменную среды.

б) В консоли администрирования SAP Commerce перейдите в Консоль вкладку и выберите Гибкий поиск вариант.

The ГибкийПоиск является страница.

в. Чтобы проверить статус одобрения всех концертов, введите запрос ВыБРАТЬ {p.pk}, {p.code}, {p.name}, {q.code} ИЗ {Концерт КАК p}, {СтатусУтвержденияСтатьиКод q} ГДЕ {p.approvalstatus} = {q.pk} в ГибкийПоиск текстовое поле запроса и щелкните Выполнить кнопка.

. Создайте задание с помощью скрипта Groovy.

а. В консоли администрирования SAP Commerce перейдите в Консоль вкладку, выберите Языки сценариев вариант и выберите Крутов Тип скрипта выпадающее меню.

б) Введите скрипт Groovy.

```
импорт de.hybris.platform.cronjob.enums.*
импорт de.hybris.platform.servicelayer.cronjob.PerformResult
импорт de.hybris.platform.servicelayer.search.*
импорт de.hybris.platform.servicelayer.model.*
импорт de.hybris.platform.catalog.enums.СтатусУтвержденияСтатьи
импорт Concerttours.model.МодельКонцерта
```

```

поискСервис = spring.getBean("flexibleSearchService")
модельСервис = spring.getBean("modelService")
query = new FlexibleSearchQuery("Выберите {pk} из {Concert}");
searchService.search(query).getResults().each {
    если (it.daysUntil < 1) {

        it.approvalStatus = СтатусУтвержденияСтатьи.ПРОВЕРКА
    }
    modelService.saveAll()
}

```

в) Сохраните скрипт в базе данных, введя имя clearoldconcerts в код поле над скриптом и нажмите кнопку Сохранить кнопка.

г. Запустите скрипт, нажав на кнопку Выполнить кнопка.

Скрипт должен работать без ошибок.

е. Нажмите на Откатить кнопку, чтобы она отображала «Принять» и снова запустите скрипт.

На этот раз результаты сохраняются в базе данных, и вы сможете увидеть их в других консолях.

Интерактивный ярлык SAP Commerce 123: если вы не можете создать скрипт Groovy самостоятельно или хотите пропустить этот шаг, выполните следующую команду.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#simulateGroovyScript test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#simulateGroovyScript test
```

. Получите список концертов с указанием статуса их одобрения еще раз.

а. В консоли администрирования SAP Commerce перейдите в Консоль вкладку и выберите Гибкий Поиск вариант.

The Гибкий Поиск появляется страница.

б) Чтобы проверить, что новый элемент ServicelayerJob был создан, введите запрос ВЫБРАТЬ {p.pk}, {p.code}, {p.name}, {q.code} ИЗ {Концерт КАК p}, {СтатусУтвержденияСтатьиКод q} ГДЕ {p.approvalstatus} = {q.pk} в Гибкий Поиск текстовое поле запроса и щелкните Выполнить кнопка.

Значения approvalStatus изменились.

. Оберните задачу в сценарий задачу и вызовите ее с помощью триггера, запустив скрипт ImpEx в Импекс Импорт консоль.

```

# Определить ScriptingJob
INSERT_UPDATE ScriptingJob; code[unique=true ];scriptURI ;clearoldconcertsJob;model://clearoldconcerts

# Определить CronJob
INSERT_UPDATE CronJob; code[unique=true ];job(код);sessionLanguage(isocode);clearoldconcertsCronJob;clearoldconcertsJob;en

# Определите курок
Вставить, обновление Триггер;cronjob(код)[уникальный=истина];cronExpression
# % afterEach: impex.getLastImportedItem().setActivationTime(new Date());
clearoldconcertsCronJob; 0/10 * * * * ?

```

Вывод консоли показывает, что задание запускается каждые 10 секунд. В производственной среде вы бы изменили триггер на запуск один раз в конце дня.

. Запустите testGroovyScript повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testGroovyScript test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testGroovyScript test
```

. Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add . ; git commit -m "Создание скрипта Cronjob с использованием Groovy"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Создание скрипта Cronjob с использованием Groovy"
```

## Панель управления администрированием бэк-офиса

Backoffice Administration Cockpit — это удобный пользовательский интерфейс на основе браузера для просмотра, создания и управления данными в SAP Commerce.

Backoffice Administration Cockpit состоит из виджетов, каждый из которых имеет определенную функцию. Это позволяет вам управлять рядом различных типов данных. Основные данные, используемые SAP Commerce, доступны в перспективе администрирования. Вы можете выбрать отдельные категории из дерева проводника, списка меню в левой части главного окна. Отсюда вы можете просматривать, перечислять, вводить и редактировать данные каталога и продукта, данные пользователя, включая роли и разрешения, настройки цен, интернационализацию или базовые данные конфигурации системы.

Для концертных туров Расширение Backoffice Administration Cockpit полезно для удобного просмотра и редактирования данных о группах и турках.

Родительская тема: [Изучите тип SAP Commerce](#)

Предыдущий: [Кратко](#)

Следующий: [Локализация](#)

## Сопутствующая информация

[Панель управления администрированием бэк-офиса](#)

## Установить панель администрирования Backoffice

Добавьте платформу бэк-офиса расширение вашей установки и используйте его для составления списка ваших элементов.

## Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.java.

```
общественный недействительный testBackofficeProductListingContainsTheBands () {
    loginToBackOffice();
    accessBackofficeProducts(); assertTrue(waitFor("span", "Гранд Тур -
    Монреаль"));
}
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testBackofficeProductListingContainsTheBands test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testBackofficeProductListingContainsTheBands test
```

## Процедура

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Добавьте платформа-бэк-офис расширение вашеролокалекстеншн.xml, который расположен в конфигурации папка.

```
<hybrisconfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="resources/schemas/extensions.xsd">
    <расширения>
        <!--
            Все расширения, расположенные в ${HYBRIS_BIN_DIR}/platform/ext, загружаются автоматически.
            Более подробную информацию о настройке доступных расширений можно найти здесь: https://help.sap.com/viewer/DRAFT/b490bb4e85bc
        -->
        <расширение имя="концертные туры"/>
        <расширение имя="platformbackoffice"/>

        <!-- расширенный шаблон -->
        <расширение имя="yempty" />
        <расширение имя="yhacext" />

    </расширения>
</hybrisconfig>
```

Интерактивный ярлык SAP Commerce 123: если вы не можете добавить расширение или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/config/localextensions.xml с содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/withbackoffice/localextensions.xml.

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/withbackoffice/localextensions.xml $HYBRIS_HOME_DIR/hybris/config/localextensions.xml
```

```
копировать /у %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\withbackoffice\localextensions.xml %HYBRIS_HOME_DIR%\hybris\config\localextension
```

. Перестройте систему.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; сборка ant
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и сборка ant
```

. Переинициализируйте систему.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; инициализация ant
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и инициализация ant
```

. Запустите SAP Commerce.

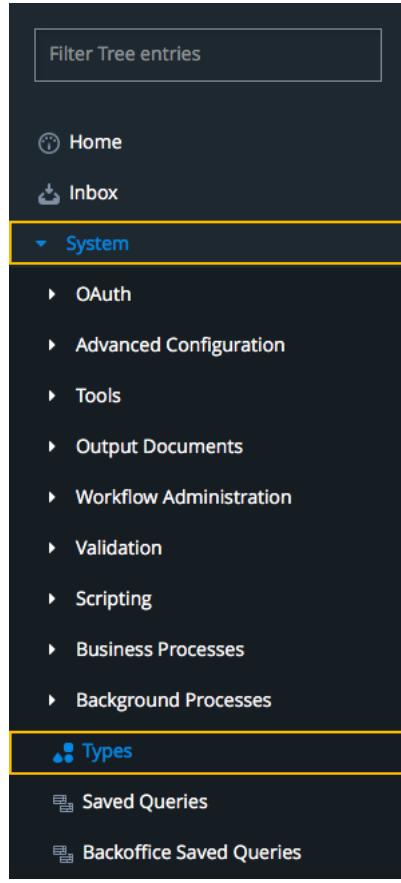
```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh start
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и запустить hybrisserver.bat
```

. Изучите панель управления бэк-офисом.

а. Открыть бэк-офис по адресу <https://localhost:9002/backoffice> войдите в систему как пользователь **админи** пароль, который вы определили как переменную среды.

б. В дереве проводника Backoffice Administration Cockpit перейдите к **Типы систем**.



в. В поиске, введите группу. Нажмите на группу элемента в результатах поиска, чтобы просмотреть подробную информацию о типе элемента.

**Band**

SEARCH

6 items

Identifier	Extensionname
Band	concerttours
Band2MusicType	concerttours
Band2MusicTypebandsColl	concerttours

**[Band]**

REFRESH SAVE

COMMON PROPERTIES XML REPRESENTATION EXTENDED ADMINISTRATION

**ESSENTIAL**

**PROPERTIES**

Attributes defined for this type ?

Qualifier	Feature type	Readable	Editable	Optional	Search
albumSales	java.lang.Long [java.lang.Long]	true	true	true	true
code	java.lang.String [java.lang.String]	true	true	true	true
history	java.lang.String [java.lang.String]	true	true	true	true
name	java.lang.String [java.lang.String]	true	true	true	true

. В дереве проводника Backoffice Administration Cockpit перейдите к Каталогу продаж для поиска концертных туров.

Вам следует ознакомиться со списком концертов.

The screenshot shows the SAP Backoffice Administration interface. At the top, there's a navigation bar with icons for administration, search, and other system functions. Below the navigation is a search bar with a magnifying glass icon and a yellow 'SEARCH' button. The main area displays a table of products:

Product.	Article Number	Identifier	Approval	Catalog version
<input type="checkbox"/>	20170105	Grand Tour - Boulder	approved	concertoursProductCatalog : Online
<input type="checkbox"/>	20170104	Grand Tour - Gliwice	approved	concertoursProductCatalog : Online
<input type="checkbox"/>	20170103	Grand Tour - Montreal	approved	concertoursProductCatalog : Online
<input type="checkbox"/>	20170102	Grand Tour - London	approved	concertoursProductCatalog : Online
<input type="checkbox"/>	20170101	Grand Tour - Munich	approved	concertoursProductCatalog : Online

A modal window is open for the product 'Grand Tour - Munich [20170101] - concertoursProductCatalog : Online'. It contains tabs for Properties, Attributes, Category System, Prices, Multimedia, Variants, Extended Attributes, and BMECAT. The 'Properties' tab is selected. The 'Essential' section shows the following details:

Article Number	Identifier	Catalog version	Approval
20170101	Grand Tour - Munich	concertoursProductCatalog : ...	approved

The 'Variants Attributes' section is also visible below.

.Запустите testBackofficeProductListingContainsTheBands повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testBackofficeProductListingContainsTheBands test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testBackofficeProductListingContainsTheBands test
```

.Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add .; git commit -m "Установить панель администрирования Backoffice"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Установить панель администрирования Backoffice"
```

## Локализация

Локализация направлена на адаптацию SAP Commerce к нескольким языкам.

Локализация встроена в SAP Commerce Platform с нуля, а не прикреплена как нечто второстепенное. В результате вы можете объявлять атрибуты и отношения типов элементов как локализованный расширение \*-элементы.xml, и система автоматически представляет несколько значений для разных локалей. Даже сами имена систем типов могут быть локализованы, так что когда имена типов элементов и атрибутов появляются в пользовательских интерфейсах, они могут быть на выбранном пользователем языке.

Когда система сборки SAP сталкивается с локализованный ключевое слово, вместо одного значимого атрибута он создает карту значений, привязанных к языку для этого атрибута, и генерирует эквивалентную конструкцию в базе данных.

Impex имеет встроенный синтаксис для указания значений для локализованных значений, и часто бывает более удобно разделять содержимое файлов impex по языку. Хотя нет необходимости использовать отдельные файлы для содержимого, специфичного для языка, обычно это значительно упрощает его понимание и управление. В частности, это упрощает добавление и удаление содержимого для определенного языка.

В интеграционных тестах hybris123 мы использовали службу модели для создания новых экземпляров групп, концертов и т. д. Служба модели обрабатывает предоставление локали для использования при работе с локализованными атрибутами. К сожалению, когда дело доходит до модульных тестов, у вас нет контекста, предоставляемого службой модели. В результате, когда вы вызываете обычный геттер или сеттер локализованного атрибута, объект модели не знает, на какой язык сопоставлять значение. Если вы обращаетесь только к локализованным атрибутам в своем тестовом коде, вы можете решить эту проблему, используя геттер и сеттер, которые задают локаль, которую вы хотите использовать, в качестве дополнительного параметра.

Если в тестируемом вами коде вызываются простые методы доступа, то нет возможности предоставить LocaleProvider для их использования. В результате вы больше не можете тестировать этот код в модульном teste. Вам придется вернуться к использованию интеграционных тестов.

Если, например, вы замените вызовы на получить Историю() и setHistory() в тесте DefaultBandFacadeUnitTest методы cgetHistory(Locale.ENGLISH) и setHistory(BAND\_HISTORY, Locale.ENGLISH), этот тест все равно не будет пройден, потому что код, который вы тестируете в реальном DefaultBandFacade класс вызывает простые получить Историю()

аксессор.

Единственный способ, которым вы можете заставить этот тест пройти, — это проверить, работаете ли вы вне контекста SAP Commerce Server, и изменить его поведение, чтобы справиться с этим. Это бессмысленно, поэтому, к сожалению, вам придется прекратить использовать этот модульный тест и положиться на более медленный интеграционный тест.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Панель управления администрированием бэк-офиса](#)

Следующий:[Локализация в административной панели бэк-офиса](#)

## Сопутствующая информация

[Интернационализация и локализация](#)

[Требования к кодировке символов для файлов](#)

[Локализации Языки и локализация](#)

## Локализуйте значения атрибутов с помощью ImpEx

Локализуйте значения атрибутов и имена систем типов.

## Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.java.

```
public void testLocalizedServiceLayerTest() {
    assertTrue(checkTestSuiteXMLMatches("(.*testsuite ошибки=\\\"2\\\" (.* имя=\"DefaultBandFacadeUnitTest\" пакет=\"concerttours.facades.impl\" тесты= checkTestSuiteXMLMatches(\".*testsuite ошибки=\\\"0\\\" неудачи=\\\"0\\\" (.*) имя=\"DefaultBandServiceUnitTest\" пакет=\"concerttours.service.impl\" te })
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testLocalizedServiceLayerTest тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testLocalizedServiceLayerTest тест
```

## Процедура

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Изменить concerttours-items.xml для включения локализацийэкштегиисторияатрибуты.

```
<квалификатор атрибута="xshstet" тип="локализованныйjava.lang.String">
<description>xshstet концертного тура для социальных сетей</description> <persistence
type="property" />
</атрибут>
```

```
<квалификатор атрибута="history" тип="localized:java.lang.String">
<description>история группы</description> <persistence
type="property" />
</атрибут>
```

Интерактивный ярлык SAP Commerce 123: если вы не можете изменить concerttours-items.xml или вы хотите пропустить этот шаг, замените `<HYBRIS_HOME_DIR>/hybris/bin/custom/concerttours/resources/concerttours-items.xml` содержанием `<HYBRIS_HOME_DIR>/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-itemsWithLocalization.xml`

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-itemsWithLocalization.xml $HYBRIS_HOME_DIR/hybris
```

```
копировать /у %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\resources\concerttours-itemsWithLocalization.xml %HYBRIS_HOME_DIR%\h
```

. Разделить концертные туры-группы.impex файл на три файла для локализации связанных файлов impex: один файл для содержимого, не зависящего от языка (/impex/concerttours-bands.impex), один для английского контента (/impex/concerttours-bands-en.impex), и один для немецкого контента (/impex/concerttours-bands-de.impex).

# ImpEx для импорта групп в магазин Little Concert Tours

```
INSERT_UPDATE Группа;код[unique=true];название;альбомПродажи ;A001;yRock;1000000
;A006;yBand;
;A003;yДжаз;7
;A004;запрещено;427
;A002;Сиркен;2000
;A005;Хор;49000
;A007;Тихо;1200
```

# ImpEx для импорта немецких атрибутов групп в магазин Little Concert Tours Store \$lang=de

```
INSERT_UPDATE Группа;код[unique=true];история[lang=$lang]
;A001;Gelegentliche Tribut-Rock-Band bestehend aus Führungskräften eines führenden Commerce-Software-Anbieters
;A006;Die niederländische Tribut-Rock-Band, die im Jahr 2013 mit klassischen Rock-Melodien aus den sechziger, siebziger Jahren ge
```

;A003;Экспериментальная джазовая группа из Лондона, живые музыкальные новости, которые можно услышать в неожиданных комбинациях и последующих выступлениях ;A004;Verjunge polnische Jungenband из 1990-х годов - dieses Genre der Popmusik am zweifelhaftesten am besten  
;A002;Группа пения а капелла mit Sitz в Мюнхене; Eine erhabende Mischung aus Traditionellen und zeitgenössischen Liedern ;A005;Enthusiastischer, lärmender Hospitalchor Singt Традиционные Евангельские песни aus dem Tiefen Süden  
;A007;Английское хоровое общество, специализирующееся на удивительных аранжировках, музыкальных мелодиях и песнях

# ImpEx для импорта англоязычного контента групп в Little Concert Tours Store \$lang=en

```
INSERT_UPDATE Группа; код[unique=true]; история[lang=$lang]
;A001; группа трибют-рока, в состав которой входят старшие менеджеры ведущего поставщика коммерческого программного обеспечения.;A006;Голландская группа трибют-рока, образованная в 2013 году и исполняющая классические рок-мелодии шестидесятых, семидесятых и восьмидесятых годов.;A003;Экспериментальная джазовая группа из Лондона, играющая множество музыкальных нот вместе в неожиданных комбинациях и последовательностях.;A004;Возрожденный польский бой-бэнд 1990-х годов — этот жанр поп-музыки в его самой сомнительной лучшей форме.
;A002;Группа пения а капелла из Мюнхена; вдохновляющее сочетание традиционных и современных песен ;A005;Восторженный, шумный госпел-хор, исполняющий традиционные госпел-песни с юга ;A007;Английское хоровое общество, специализирующееся на прекрасно аранжированных, успокаивающих мелодиях и песнях
```

Интерактивный ярлык SAP Commerce 123: если вы не можете разделить концертные туры-группы.impexle на три файла или вы хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/impex/c содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/impex/withlocalization/\*

```
mkdir -p $HYBRIS_HOME_DIR/hybris/bin/custom/concerttours/resources/impex; cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/impex/withloc
```

```
эхо | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\impex\withlocalization\concerttours-bands* %HYBRIS_HOME_DIR%\hybris\bin\cu
```

. Изменить концертные туры-yBandTour.impex включить как английские, так и немецкие хэштеги.

```
# Вставьте продукты
INSERT_UPDATE Продукт; код[unique=true]; название; группа(код); хэштег[lang=en]; хэштег[lang=de]; $supercategories; название производителя; идентификатор производителя; uni ;201701;The Grand Little Tour; A001;GrandLittle; GrossWenig; x; y; штук; Концерт
```

Интерактивный ярлык SAP Commerce 123: если вы не можете изменить концертные туры-yBandTour.impexle или вы хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/impex/c содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/impex/withlocalization/\*

```
то же самое $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/impex/withlocalization/concerttours-yBandTour.impex $HYBRIS_HOME_DIR/hybris/bin/cust
```

```
xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\impex\withlocalization\concerttours-yBandTour.impex %HYBRIS_HOME_DIR%\hybris\bin\cu
```

. Обновить ConcerttoursCustomSetup класс для загрузки трех локализованных файлов вместо стандартного, который представляет собой один нелокализованный файл.

```
@SystemSetup(тип = SystemSetup.Type.PROJECT)
    публичное логическое значение addMyProjectData()
{
    LOG.info("Начало загрузки данных пользовательского проекта для расширения Concerttours");
    impexImport("/impex/concerttours-bands.impex");
    impexImport("/impex/concerttours-bands-en.impex"); impexImport(
    "/impex/concerttours-bands-de.impex"); impexImport("/impex/
    concerttours-yBandTour.impex");
    LOG.info("Загрузка данных пользовательского проекта для расширения Concerttours завершена."); return
    true;
}
```

Интерактивный ярлык SAP Commerce 123: если вам не удается обновить ConcerttoursCustomSetup класс или вы хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/setup/ConcerttoursCustomSetup.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/setup/LocalizedConcerttoursCustomSetup.java

```
то же самое $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/setup/LocalizedConcerttoursCustomSetup.java $HYBRIS_HOME
```

```
echo a | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\setup\LocalizedConcerttoursCustomSetup.java
```

. BDefaultBandServiceUnitTest класс, измените вызов на setHistory() метод для включения ссылки на требуемую локаль, чтобы модульный тест учитывал локаль.

```
bandModel.setHistory(BAND_HISTORY, Locale.ENGLISH);
```

. Добавлять импорт java.util.Locale; в список импорта BDefaultBandServiceUnitTest.java тест.

```
упаковка concerttours.service.impl;
импорт static org.junit.Assert.assertEquals;
импорт static org.mockito.Mockito.mock;
импорт static org.mockito.Mockito.when;
импорт de.hybris.bootstrap.annotations.UnitTest; java.util.Arrays;
импорт
импорт java.util.Коллекции;
импорт java.util.List;
импорт org.junit.До;
импорт org.junit.Test;
импорт концертные туры.daos.BandDAO;
импорт концертные туры.модель.BandModel;
импорт java.util.Локаль;
```

Интерактивный ярлык SAP Commerce 123: если вы не можете изменить тест или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/testsrc/concerttours/service/impl/DefaultBandServiceUnitTest.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/service/impl/Localized\_DefaultBandServiceUnitTe

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/service/impl/Localized_DefaultBandServiceUnitTest.jav
```

```
копировать /у %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\testsrc\concerttours\service\impl\Localized_DefaultBandServiceUnitTe
```

. Перестройте и повторно инициализируйте базу данных.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ant очистить все инициализировать yunitinit
```

12/3/24, 11:01 утра

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и ant clean all инициализирует yunitinit
```

. Запустите тесты и обратите внимание, что DefaultBandFacadeUnitTest терпит неудачу, но DefaultBandServiceUnitTest пропускает по причинам, описанным в обсуждении локализации.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ant alltests -Dtestclasses.packages="concerttours.*"
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и ant alltests -Dtestclasses.packages=concerttours.*
```

. Запустите тест LocalizedServiceLayerTest повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testLocalizedServiceLayerTest тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testLocalizedServiceLayerTest тест
```

. Удалить файл DefaultBandFacadeUnitTest.java.

The DefaultBandFacadeUnitTest больше несовместимо и приведет к сбою по причинам, описанным в обзоре локализации.

```
rm $HYBRIS_HOME_DIR/hybris/bin/custom/concerttours/testsrc/concerttours/facades/impl/DefaultBandFacadeUnitTest.java
```

```
дель %HYBRIS_HOME_DIR%\hybris\bin\custom\concerttours\testsrc\concerttours\facades\impl\DefaultBandFacadeUnitTest.java
```

. Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add .; git commit -m "Локализация значений атрибутов с помощью ImpEx"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Локализация значений атрибутов с помощью ImpEx"
```

. Запустите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh start
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и запустить hybrisserver.bat
```

## Локализация в административной панели бэк-офиса

Вы можете определить локализованные значения для атрибутов типа элемента непосредственно в панели администрирования Backoffice.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Локализация](#)

Следующий:[Проверка](#)

### Локализуйте значения атрибутов с помощью Backoffice

Представьте локализованные строковые записи для истории поле элементов типа группы.

#### Предпосылки

В ваш комплект документов включен следующий приемочный тест Hybris123Tests.java.

```
public void testBackofficeLocalization() выдает исключение {
    canLoginToHybrisCommerce(); navigationTo("https://localhost:9002/
    platform/init");
   waitForThenClickButtonWithText("Инициализировать");
    waitForThenClickOkInAlertWindow();
    waitForInitToComplete();
    закрытьБраузер();

    loginToBackOffice("Deutsch");
    waitForThenClickMenuItem("System");

    если (VersionHelper.getVersion().равно(Version.V2211))
        waitForThenScrollToThenClick("//span[text()='Typen']");
    еще
        waitForThenClickMenuItem("Typen");

    searchForConcertInBackoffice(); waitForThenAndClickSpan("Концерт");
    waitForThenAndClickSpan("Eigenschaften"); waitForThenClickDotsBySpan("daysUntil");
    waitForThenAndClickSpan("Подробнее", "Редактировать детали");
    AssertTrue( waitForValue("input", "Tage bis es stattfindet") );
}
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testBackofficeTect локализации
```

cd %HYBRIS\_HOME\_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris.runtime.tests.Hybris123Tests#testBackofficeTest локализации

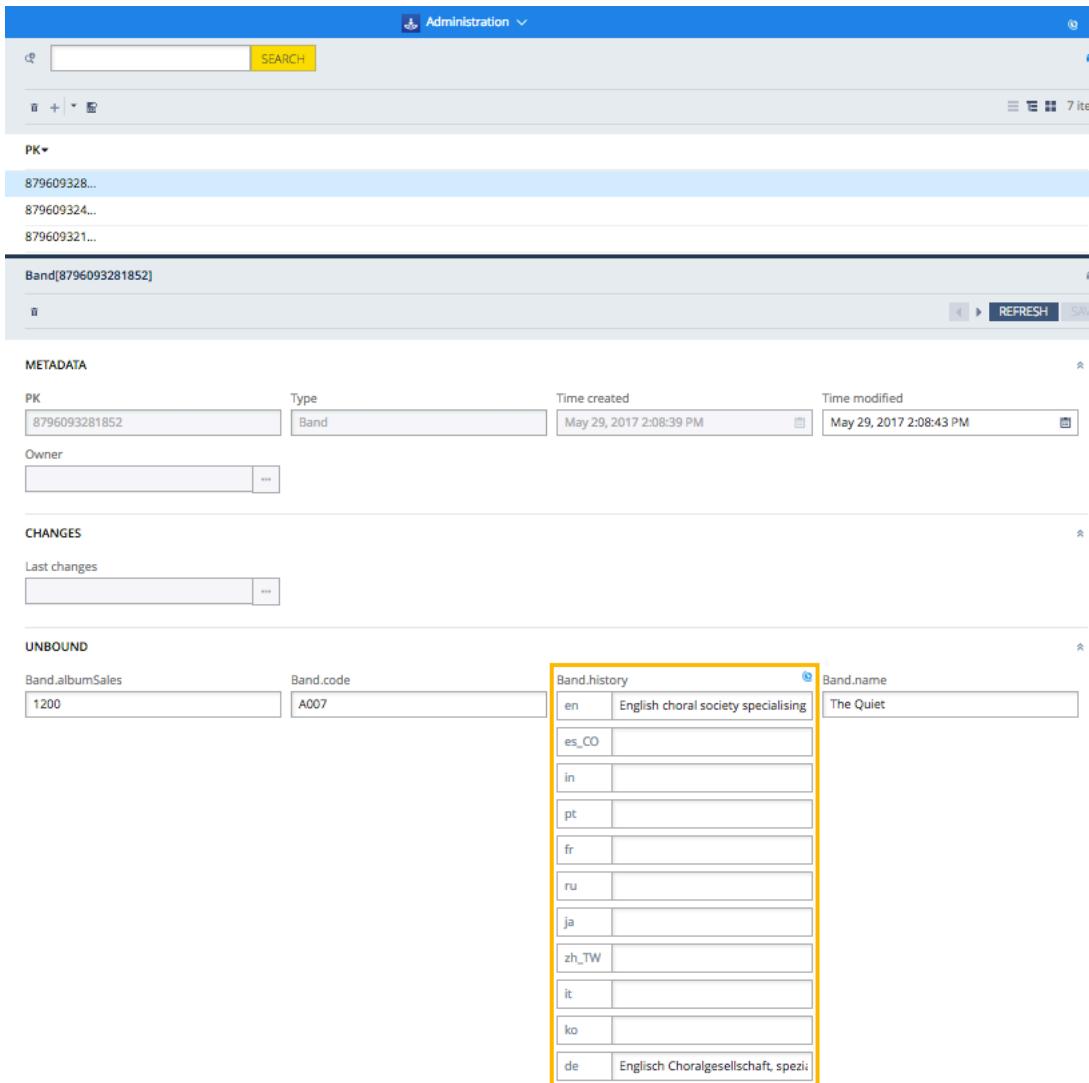
## Процедура

. Откройте бэк-офис в <https://localhost:9002/backoffice> войдите в систему как пользователь админ и пароль, который вы определили как переменную среды.

. В дереве проводника Backoffice Administration Cockpit перейдите к Типы | систем. >  .

. Поиск Группатип элемента и на панели инструментов панели сведений щелкните значок Поиск по типу|кнопка  .

. Выберите один из элементов Band и изучите его локализованные значения истории, нажав на История группы|иконка мира.



en	English choral society specialising
es_CO	
in	
pt	
fr	
ru	
ja	
zh_TW	
it	
ko	
de	Englisch Choralgesellschaft, spezi

. Добавьте локализованные записи строк в concerttours/ресурсы/локализация/concerttours-locales\_en.properties локализация системы типа le, и concerttours/ресурсы/локализация/concerttours-locales\_de.properties файл локализации системы типов, предоставляющий локализованную версию Системы типов.

Каждая запись в формате ключ=локализованное значение, следующее:

```
type.Product.hashtag.name=хэштег
type.Product.hashtag.description=тер социальных сетей
```

```
тип.Band.name=Группа
тип.Band.description=группа или коллектив музыкантов или певцов
тип.Band.code.name=код
тип.Band.code.description=的独特ый короткий идентификатор для
группы
тип.Band.name.name=имя
тип.Band.name.description=название группы
тип.Band.history.name=история
тип.Band.history.description=краткая история о группе
тип.Band.albumSales.name=продажи альбомов
тип.Band.albumSales.description=количество проданных альбомов
```

```
тип.Concert.name=Концерт
тип.Concert.description=выступление группы
тип.Concert.venue.name=место проведения
тип.Concert.venue.description=где проводится
тип.Concert.date.name=дата
тип.Concert.date.description=когда проводится
тип.Concert.concertType.name=тип концерта
тип.Concert.concertType.description=тип концерта
тип.Concert.daysUntil.name=дней до
тип.Concert.daysUntil.description=количество дней до проведения
```

```
тип.News.имя=Новости
тип.News.description=новости о группе
тип.News.date.name=дата
тип.News.date.description=когда появилась новость
тип.News.headline.name=заголовок
тип.News.headline.description=заголовок для новостного сообщения
```

type.News.content.name=контент type.News.content.description=контент для новостного элемента

type.Product.hashtag.name=хэштег  
type.Product.hashtag.description=Ter социальных сетей

type.Band.name=Группа  
type.Band.description=Группа или группа музыкантов или музыкантов type.Band.code.name=код

type.Band.code.description=Идентификатор индивидуального курса для браслета  
type.Band.name.name=Имя  
type.Band.name.description=Имя группы type.Band.history.name=История  
type.Band.history.description=Название группы  
type.Band.albumSales.name=Albumverkauf

type.Band.albumSales.description=Anzahl der verkauften Alben

type.Concert.name=Концерт type.Concert.description=Eine Aufführung von einer Band type.Concert.venue.name=Schauplatz

type.Concert.venue.description=Wo es stattfindet  
type.Concert.date.name=Датум  
type.Concert.date.description=Что такое концерт  
type.Concert.concertType.name=Концерт  
type.Concert.concertType.description=Art type.Concert.daysUntil.name=Tage bis  
type.Concert.daysUntil.description=Tage bis es stattfindet

type.News.name=Новости type.News.description=Новости с максимальной частотой type.News.date.name=Датум

type.News.date.description=Als die Nachricht angekündigt wurde  
type.News.headline.name=Überschrift  
type.News.headline.description=Schlagzeile für die Nachricht  
type.News.content.name=Inhalt  
type.News.content.description=Inhalt für die Nachricht

Если вы определяете новые типы элементов, атрибуты, отношения и т. д., вам следует предоставить локализованные версии их имен, которые могут использоваться приложениями Backoffice, такими как Backoffice Administration Cockpit.

Интерактивный ярлык SAP Commerce 123: если вы не можете добавить записи или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/localization/concerttours-locales\_en.propertiesc содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/resources/localization/concerttours-locales\_en.localizedproperties, и <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/localization/concerttours-locales\_de.propertiesc содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/resources/localization/concerttours-locales\_de.localizedproperties.

```
то же самое $HYBRIS_HOME_DIR$/hybris123/src/main/webapp/resources/concerttours/resources/localization/concerttours-locales_en.localizedproperties $HYBRIS_HOME_DIR$/hybris123/src/main/webapp/resources/concerttours/resources/localization/concerttours-locales_de.localizedproperties $HYB
```

```
echo a | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\resources\localization\concerttours-locales_en.localizedprope echo a | xcopy %HYBRIS_HOME_DIR%\\hybris123\src\main\webapp\resources\concerttours\resources\localization\concerttours-locales_de.localizedprope
```

Войдите в SAP Commerce Administration Cockpit и обновите систему с помощью новых локализаций из Обновление платформы экран. Чтобы сэкономить время, выберите только Локализовать типы вариант. Отмените выбор всего остального.



. Изучите локализованные типы в Backoffice Administration Cockpit.

а) Откройте Backoffice и в дереве проводника перейдите к Система ➤ Типы.

б) Поиск Концерт, затем нажмите на Характеристики вкладка.

в) Нажмите на три точки рядом с элементом и выберите Изменить детали для изучения деталей недвижимости. Обратите внимание на локализованное название и описание.

Concert [Concert]

GENERAL PROPERTIES XML REPRESENTATION EXTENDED ADMINISTRATIVE

**PROPERTIES**

Attributes defined for this type ?

Qualifier	Feature type
concertType	[ConcertType]
date	java.util.Date [java.util.Date]
daysUntil	java.lang.Long [java.lang.Long]
venue	java.lang.String [java.lang.String]

г. Повторите последний шаг, на этот раз ищите группу, затем снова для Новости.

Выйдите из Backoffice Administration Cockpit и войдите снова, выбрав Deutsch в качестве языка. Обратите внимание, что предоставленные немецкие значения теперь используются для имен и описания.

Commerce

Filter Tree entries

Startseite Posteingang System OAuth Erweiterte Konfiguration Tools Output Dokumente Workflow-Verwaltung Validierung Scripting Geschäftsprozesse Hintergrundprozesse Typen Gespeicherte Anfragen Gespeicherte Abfragen Backoffice Berichtsdefinitionen Personalisierung Prüfung von Systemeinrichtung Katalog Multimedia Benutzer Bestellung Preis-Einstellungen Internationalisierung Marketing SAVED QUERIES

Concert

SEARCH

Edit item Konzert [Concert] -> Tage bis [daysUntil]

Erstellungszeit Änderungszeit  
29.05.2017 15:13:46 29.05.2017 15:13:46

Besitzer Konzert [Concert]

ÄNDERUNGEN Letzte Änderungen

NICHT ZUGEWIESEN Attribut-Handler Einschränkungsgruppen  
concertDaysUntilAttributeHandler

Datenbankspalte Deklariert in Konzert [Concert]

Standardwert Beschreibung  
Tage bis es stattfindet

Nicht kopieren Eingeschließer Typ  
 True  False Konzert [Concert]

Verschlüsselt Für UI ausgeblendet  
 True  False

. Запустите тест Backoffice/локализация повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR\hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testBackofficeTest локализации
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testBackofficeTest локализации
```

. Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR\hybris; git add .; git commit -m "Локализация значений атрибутов с помощью Backoffice"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Локализация значений атрибутов с помощью Backoffice"
```

## Проверка

Фреймворк проверки данных SAP Commerce обеспечивает чистые, правильные и полезные данные. Фреймворк проверки основан на спецификации проверки Java, JSR 303. Он предлагает простой и расширяемый способ проверки данных перед их передачей на уровень сохранения.

Структура проверки данных преследует несколько целей:

- Предложить структуру для определения ограничений проверки данных простым и интуитивно понятным способом.
- Для проверки данных перед их сохранением
- Уведомлять о любых нарушениях валидации, если они произойдут

Логика проверки может быть запущена следующими способами:

- Неявно с ПроверкаПерехватчик который подключается к вызовам сохранять метод в модели
- Явно, вручную вызывая подтвердить Метод Служба проверки, и передача модели SAP Commerce или POJO для проверки

При обнаружении нарушений валидации они представляются вызывающей стороне для разрешения. Структура валидации не распространяется на выполнение клиентской валидации. Валидация происходит только на стороне сервера.

Основными компонентами ограничений проверки данных являются:

- Служба проверки: управляет ограничениями проверки и проверяет данные
- Панель администрирования бэк-офиса: предоставляет интерфейс для управления типами ограничений проверки экземпляров
- Интеграция с кабиной пилота: предоставляет пользователям обратную связь по проверке в кабине пилота.

Для работы проверки вы определяете ограничения. В следующей процедуре вы определяете ограничения проверки в элементы.xml, но есть и другие способы определения ограничений. Вы можете использовать файл ImpEx, который позволяет автоматизировать процесс и быстро добавлять дополнительные ограничения. ImpEx — идеальный способ работы между несколькими платформами. Вы также можете определять ограничения в Backoffice Administration Cockpit. Таким образом, вы можете обновлять ограничения проверки вручную и во время выполнения. Чтобы узнать больше об этих процедурах, см. [Ограничения проверки в бэк-офисе](#).

Родительская тема: [Изучите тип SAP Commerce](#)

Предыдущий: [Локализация в административной панели бэк-офиса](#)

Следующий: [Ограничения проверки в бэк-офисе](#)

Сопутствующая информация

[Структура проверки данных](#)

[Служба проверки в Backoffice Framework](#)

## Определить ограничения проверки в файле Items.xml

Определите ограничение проверки в элементы.xml.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест Hybris123Tests.java.

```
public void testValidationConstraintViaItemsXml() {
    loginToBackOffice();
    waitForThenClickMenuItem("Система");
    если (VersionHelper.getVersion().равно(Version.V2211))
        waitForThenScrollToThenClick("//span[text()='Типы']");
    еще
        waitForThenClickMenuItem("Типы");
    waitForThenDoBackofficeSearch("Полоса");
    waitForThenClick("диапазон", "Полоса");
    waitForImageWithTitleThenClick("Поиск по типу");
    waitForThenDoBackofficeSearch("");
    waitForThenClick("span", "Тишина");

    waitForThenUpdateInputField("Тишина", "Хор");
    assertTrue(waitForThenClick("кнопка", "Сохранить"));
}
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testValidationConstraintViaItemsXml test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testValidationConstraintViaItemsXml test
```

### Процедура

Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

12/3/24, 11:01 утра

. Сделайте названия групп обязательными и уникальными, добавив следующие модификационные теги в имя атрибут определения типа элемента Band в concerttours-items.xml из ресурсов как концептные туры расширение.

```
<квалификатор атрибута="имя" type="java.lang.String"> группа</
    <description>имя
        <тип сохранения>свойство
    </тип сохранения>
    <модификаторы необязательные="false" unique="true" />
</атрибут>
```

Интерактивный ярлык SAP Commerce 123: если вы не можете редактировать concerttours-items.xml или вы хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttours-items.xml с содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-itemsWithValidation.xml

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-itemsWithValidation.xml $HYBRIS_HOME_DIR/hybris/bin
```

```
копировать /у %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\resources\concerttours-itemsWithValidation.xml %HYBRIS_HOME_DIR%\hyb
```

. Перестройте и инициализируйте SAP Commerce с помощью Ant.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; инициализация сборки ant
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и инициализация сборки ant
```

. Протестируйте ограничения проверки в Backoffice.

а. Запустите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh start
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и запустить hybrisserver.bat
```

б. Войти в систему <https://localhost:9002/backoffice> используя логин admin и пароль, который вы определили как переменную среды.

в. Выберите **Типы системы** узлы в дереве проводника слева.

г. Найдите элемент Band Type и на панели инструментов подробностей щелкните значок **Поиск по типу** икона.

е. Выберите один из элементов Band и попробуйте удалить значение его атрибута name, удалив содержимое поля name и нажав кнопку Сохранять кнопка.

Появится сообщение о технической ошибке, и операция сохранения будет отклонена.

е. Попробуйте изменить атрибут имени одной группы на то же имя, что и у другой группы.

Появится сообщение о технической ошибке, и операция сохранения будет отклонена.

. Запустите testValidationConstraintViaItemsXml повторите приемочное испытание и подтвердите, что оно теперь прошло.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testValidationConstraintViaItemsXml test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testValidationConstraintViaItemsXml test
```

. Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add .; git commit -m "Определить ограничения проверки в файле Items.xml"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Определить ограничения проверки в файле Items.xml"
```

## Ограничения проверки в бэк-офисе

Вы можете создавать и определять ограничения проверки в панели администрирования SAP Commerce Backoffice.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Проверка](#)

Следующий:[Ограничения проверки в ImpEx](#)

### Определите ограничение проверки с помощью Backoffice

Определите ограничение проверки в Backoffice.

#### Предпосылки

В ваш комплект документов включен следующий приемочный тест Hybris123Tests.java:

```
public void testValidationConstraintViaBackoffice() { loginToBackOffice();

selectConstraintsPage();
tryToViolateTheNewConstraint();
assertTrue(waitFor("span", "Продажи альбомов не должны быть отрицательными")); }
```

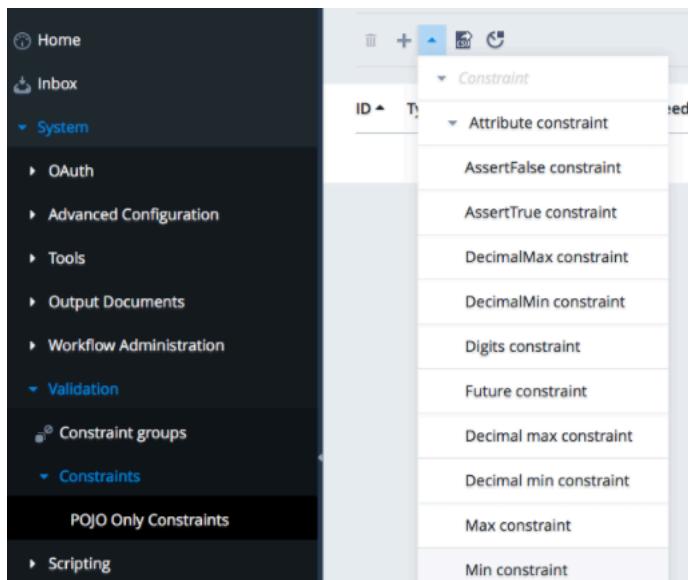
Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testValidationConstraintViaBackoffice test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris.runtime.tests.Hybris123Tests#testValidationConstraintViaBackoffice test
```

## Процедура

- . В административной панели бэк-офиса откройте **Администрация** перспектива панели проводника.
- . Перейдите к **Ограничения проверки системы**.
- . Создайте новое ограничение минимального значения.
  - а. Нажмите стрелку меню справа от кнопки «+» на панели инструментов, чтобы раскрыть иерархический список типов ограничений.
  - б. Расширьте **Ограничение атрибута** пункт меню, нажав на стрелку слева от текста пункта меню.
  - в. Нажмите на **Минимальный элемент ограничения для отображения** для создания нового минимального ограничения волшебник.



- г. Дайте новому ограничению идентификатор **newОграничение**.
- е. Установите **Минимальное значение** и установите значение **0**, чтобы запретить отрицательные значения.
- е. Поскольку вы хотите проверить тип вложения, начните вводить **Группа** в **Составной тип для проверки** поле и выбрать **Группа** из списка предложений, когда он появится.
- г. Сделайте то же самое в **Дескриптор атрибута для проверки** поле для выбора альбома **Продажи** атрибут.
- ч. Щелкните **Следующий**.

**Create New Min constraint**

PROVIDE ALL MANDATORY FIELDS > ADDITIONAL ATTRIBUTES

ID: <input type="text" value="AlbumSalesMustNotBeNegative"/>	
Annotation class: <input type="text" value="javax.validation.constraints.Min"/>	
Minimal value: <input type="text" value="0"/>	
Enabled: <input checked="" type="radio"/> True <input type="radio"/> False	
Enclosing Type: <input type="text" value="Band [Band]"/>	
Attribute descriptor to validate: <input type="text" value="Band [Band] -&gt; album sales [albumSales]"/>	
Validation languages: <input type="text"/> ***	
<input type="button" value="CANCEL"/> <input type="button" value="NEXT"/> <input style="background-color: yellow; border: 1px solid black; color: black; font-weight: bold;" type="button" value="DONE"/>	

- и. Введите сообщение об ошибке **Продажи альбомов должны быть > 0. Обратите внимание.** в **Ошибка** поле сообщения.
- ж. Необязательно: Нажмите на символ глобуса, чтобы получить доступ к другим языковым параметрам, чтобы вы могли ввести сообщение об ошибке на разных языках.
- к. Нажмите на **Сделанный**.

## Create New Min constraint

PROVIDE ALL MANDATORY FIELDS > ADDITIONAL ATTRIBUTES

Severity:

Error message:

I. Нажмите самую правую кнопку на панели инструментов, чтобы перезагрузить механизм проверки, чтобы он учел новое ограничение.



Интерактивный ярлык SAP Commerce 123: Если вы не можете создать новое ограничение в Backoffice самостоятельно или хотите пропустить этот шаг, выполните следующий тест. Этот тест создает ограничение.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testCreateValidationConstraintViaBackoffice test
```

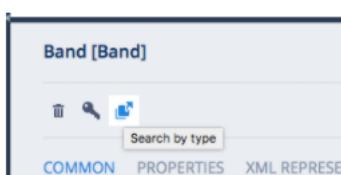
```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testCreateValidationConstraintViaBackoffice test
```

. Проверьте, работает ли новое ограничение.

а. Перейдите в **Клипы систем** в панели проводника.

б) Найдите тип элемента Band.

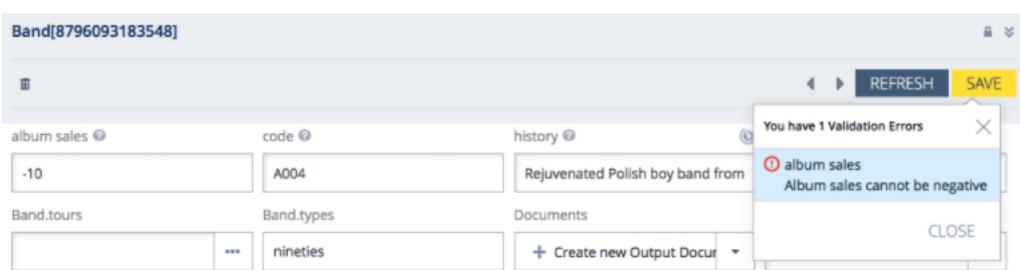
в. Нажмите кнопку **Поиск** по типу/написанию на панели инструментов для вывода списка групп в базе данных.



г. Выберите одну из групп.

е. Попробуйте установить атрибут продаж альбома на отрицательное число в панели сведений и нажмите Сохранить.

е. Убедитесь, что в списке ошибок проверки отображается правильное сообщение об ошибке.



. Запустите **testValidationConstraintViaBackoffice**, повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testValidationConstraintViaBackoffice test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testValidationConstraintViaBackoffice test
```

. Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add .; git commit -m "Определение ограничения проверки с помощью Backoffice"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Определение ограничения проверки с помощью Backoffice"
```

#### Следующие шаги

В следующем разделе вы изучите другой способ создания этого ограничения. Поэтому перед продолжением удалите созданное вами ограничение.

## Ограничения проверки в ImpEx

Вы можете определить ограничения проверки в файлах ImpEx, что упрощает повторную загрузку ограничений после инициализации базы данных.

Ограничения проверки сохраняются в базе данных. Если вы часто повторно инициализируете базу данных во время разработки и тестирования проекта, существует опасность, что вы можете стереть любые сохраненные ограничения. Чтобы облегчить усилия по восстановлению ограничений проверки, хорошей идеей будет определить и загрузить их с помощью файлов ImpEx.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Ограничения проверки в бэк-офисе](#)

Следующий:[Пользовательские ограничения проверки](#)

## Определите ограничение проверки с помощью ImpEx

Определите и загрузите ограничения проверки с помощью файлов ImpEx.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.java.

```
public void testValidationConstraintAfterImpex() {
    loginToBackOffice();
    изменитьABandToHaveNegativeAlbumSales();
    если (VersionHelper.getVersion().equals(Version.V2205) || VersionHelper.getVersion().equals(Version.V2211))
       waitFor("div", "У вас 1 сообщение проверки");
    еще
        waitFor("div", "У вас 1 ошибка проверки");
    assertTrue(waitFor("span", "Продажи альбома не должны быть отрицательными"));
}
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testValidationConstraintAfterImpex test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testValidationConstraintAfterImpex test
```

### Процедура

. Создайте новый файл ImpEx, назовите его `essentialdata-ограничения.impex`, вресурсы/импех папка концертные туры расширение.

```
INSERT_UPDATE MinConstraint;id|unique=true;severity;code,itemtype(code);active;аннотация;дескриптор(enclosingType(code),квалификатор);message[]:AlbumSalesMustNotBeNegative;ERROR;Severity,true;javax.validation.constraints.Min;Band.albumSales;Albumverkäufe dürfen nicht negativ sein;Album
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать файл ImpEx или хотите пропустить этот шаг, скопируйте `/hybris123/src/main/webapp/resources/impex/validation/essentialdata-constraints.impex` /hybris/bin/custom/concerttours/resources/impex/

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/impex/validation/essentialdata-constraints.impex $HYBRIS_HOME_DIR/hybris/bin/custom/con
```

```
копировать /у %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\impex\validation\essentialdata-constraints.impex %HYBRIS_HOME_DIR%\hybris\bin\cus
```

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Инициализируйте базу данных.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; инициализация ant
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и инициализация ant
```

. Запустите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh start
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и запустить hybrisserver.bat
```

. Подтвердите, что вы не можете спасти группу с отрицательными продажами альбомов.

а. Перейдите в [Клипы систем панели проводника](#).

б) Найдите тип элемента Band.

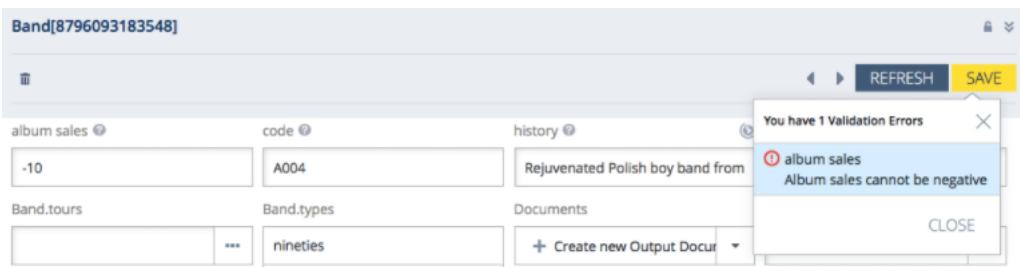
в. Нажмите кнопку [Поиск по типу](#) на панели инструментов для вывода списка групп в базе данных.



г. Выберите одну из групп.

е. Попробуйте установить атрибут продаж альбома на отрицательное число в панели сведений и нажмите Сохранить.

е. Убедитесь, что в списке ошибок проверки отображается правильное сообщение об ошибке.



. Запустите [testValidationConstraintAfterImpex](#) повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testValidationConstraintAfterImpex test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testValidationConstraintAfterImpex test
```

. Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR\hybris; git add .; git commit -m "Определение ограничения проверки с помощью ImpEx"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Определение ограничения проверки с помощью ImpEx"
```

## Пользовательские ограничения проверки

Хотя фреймворк проверки предоставляет все ограничения из спецификации JSR 303, иногда вам могут потребоваться другие типы ограничений, специфичные для вашего проекта.

История группы часто устанавливается как часть текста-заполнителя вместо значимого, подлинного фона группы. Создайте ограничение, которое предотвращает это явление, ища текст, который отсутствует или начинается с " lorem ipsum".

Новый тип ограничения расширяет АтрибутОграничениепотому что вы проверяете значения атрибутов, а не типа в целом. Если вы хотите создать ограничения, которые применяются ко всему элементу, расширьте Ограничение типавместо этого. Только Атрибут содержит ссылку на класс аннотации Java, который определяет наше ограничение как аннотацию Java.

@Ограничениеаннотация вид [aNotLoremIpsumValidator](#) класс, который предоставляет фактическую логику для проверки ограничения. Ограничение проверяет, что значение не равно null или пусто, и что оно не начинается с фразы lorem ipsum. Конечно, вы можете пойти более сложным путем, но этого достаточно для целей упражнения.

Обратите внимание, что вы можете увидеть другие свойства сопоставления для встроенных типов ограничений в файле `ext/validation/project.properties` в проекте платформы.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Ограничения проверки в ImpEx](#)

Следующий:[Интеграционный тест для пользовательского ограничения](#)

## Определить пользовательское ограничение проверки

Хотя фреймворк проверки обеспечивает все ограничения из спецификации JSR 303, иногда вам могут потребоваться другие типы ограничений.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест `Hybris123Tests.java`.

```
public void testValidationCustomConstraint()
{
    loginToBackOffice();
    selectConstraintsPage();
    deleteExistingMinConstraint("NotIpsum");
    addNewCustomConstraint("NotIpsum");
    reloadConstraints();
    попробуйтеViolateTheNewCustomConstraint();
    AssertTrue(waitFor("span", "No Lorem Ipsum"));
}
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testValidationCustomConstraint тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testValidationCustomConstraint тест
```

## Процедура

. Создайте тип элемента ограничения.

Добавьте следующее определение типа элемента в concerttours-items.xml ле в ресурсах концептные туры расширение.

```
<itemtype code="NotLoremIpsumConstraint" extends="AttributeConstraint"
автосоздание="истина" генерировать="истина">
<description>Пользовательское ограничение, проверяющее наличие текста Lorem Ipsum.</description>
<attributes>
    <калификатор атрибута="аннотация" тип="java.lang.Class">
        переобъявить="истина"
        <модификаторы write="false" initial="true" необязательно="false" /> <значение
        по умолчанию>
            concerttours.constraints.NotLoremIpsum.class </
            defaultvalue>
        </атрибут>
    </атрибуты>
</тип_элемента>
```

Интерактивный ярлык SAP Commerce 123: если вы не можете редактировать concerttours-items.xml или вы хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttours-items.xml с содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-itemsWithCustomValidation.xml

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-itemsWithCustomValidation.xml $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-itemsWithCustomValidation.xml >%HYBRIS_HOME_DIR%
```

```
типа %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\resources\concerttours-itemsWithCustomValidation.xml > %HYBRIS_HOME_DIR%
```

. Создайте интерфейс аннотаций под названием NotLoremIpsum в пакете под названием концептные туры.ограничения под источником концептные туры расширение.

```
упаковка концептные туры.ограничения;
импорт static java.lang.annotation.ElementType.FIELD;
импорт static java.lang.annotation.RetentionPolicy.RUNTIME;
импорт java.lang.annotation.Documented;
импорт java.lang.annotation.Retention;
импорт java.lang.annotation.Target;
импорт javax.validation.Constraint;
импорт javax.validation.Payload;

{@Цель(
{ ПОЛЕ })
@Retention(BРЕМЯ ВЫПОЛНЕНИЯ)
@Constraint(проверено = NotLoremIpsumValidator.класс)
@Documented
публичный @interface NotLoremIpsum {

    Стока message() по умолчанию "<concerttours.constraints.NotLoremIpsum.message>"; Класс<?>[]
    groups() по умолчанию {};
    Класс<? extends Payload>[] payload() по умолчанию {};
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать интерфейс или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/constraints/NotLoremIpsum.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/constraints/NotLoremIpsum.java

```
то же самое $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/constraints/NotLoremIpsum.java $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/constraints/NotLoremIpsum.java >%HYBRIS_HOME_DIR%
```

```
echo f | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\constraints\NotLoremIpsum.java %HYBRIS_HOME_DIR%
```

. Создайте класс валидатора, создав новый класс с именем NotLoremIpsumValidator в том же самом концептных туров ограничениях пакет под источником концептные туры расширение и копирование следующего содержания.

```
упаковка concerttours.constraints;
импорт javax.validation.ConstraintValidator;
импорт javax.validation.ConstraintValidatorContext;

открытый класс NotLoremIpsumValidator реализует ConstraintValidator<NotLoremIpsum, String> {

    @Переопределить
    public void initialize(final NotLoremIpsum constraintAnnotation) {
        // пустой
    }
    @Переопределить
    public boolean isValid(конечное строковое значение, конечный контекст ConstraintValidatorContext) {
        возвращаемое значение != null && !value.isEmpty() && !value.toLowerCase().startsWith("lorem ipsum");
    }
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать пользовательский валидатор или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/constraints/NotLoremIpsumValidator.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/constraints/NotLoremIpsumValidator.java

```
то же самое $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/constraints/NotLoremIpsumValidator.java $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/constraints/NotLoremIpsumValidator.java >%HYBRIS_HOME_DIR%
```

```
echo f | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\constraints\NotLoremIpsumValidator.java %HYBRIS_HOME_DIR%
```

. Остановите SAP Commerce.

12/3/24, 11:01 утра

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Перестройте и инициализируйте базу данных, поскольку вы изменили элементы.xml в.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; инициализация сборки ant
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и инициализация сборки ant
```

. Добавьте следующее свойство к местные.свойстве в <HYBRIS\_CONFIG\_DIR> каталог, чтобы сообщить платформе, что ваше ограничение предназначено для проверки строк символов (а не для чисел и дат).

```
проверка.ограничения.атрибут.сопоставление.концертных.тuros.ограничения.NotLoremIpsum=строки
```

Интерактивный ярлык SAP Commerce 123; если вы не можете редактировать местные.свойстве или вы хотите пропустить этот шаг, выполните следующую команду.

```
cat $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/resources/config/validation.properties >> $HYBRIS_HOME_DIR/hybris/config/
```

```
тип %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\resources\config\validation.properties >> %HYBRIS_HOME_DIR%\hybris\conf
```

. Запустите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh start
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и запустить hybrisserver.bat
```

. Проверьте новый тип ограничения.

а. Открыть бэк-офис по адресу <https://localhost:9002/backoffice> войдите в систему как пользователь админ пароль, который вы определили как переменную среды.

б) Создать новый NotLoremIpsum ограничение для атрибута истории диапазона таким же образом, как вы создали новый МинОграничение предварительно и дать ему идентификатор NotIpsum.

в. Нажмите самую правую кнопку на панели инструментов, чтобы перезагрузить механизм проверки, чтобы он учел новое ограничение.



г. Попробуйте изменить атрибуты истории одной из групп так, чтобы она начиналась словом ipsum убедитесь, что при нажатии появляется правильное сообщение об ошибке Сохранять.

. Запустите testValidationCustomConstraint повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testValidationCustomConstraint тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testValidationCustomConstraint тест
```

. Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add . ; git commit -m "Определить пользовательское ограничение проверки"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Определить пользовательское ограничение проверки"
```

## Интеграционный тест для пользовательского ограничения

Ознакомьтесь с тем, как использовать службу проверки SAP Commerce в коде.

Напишите тест, который проверяет, работает ли ограничение так, как ожидалось. Используйте ImpEx для загрузки ограничения в базу данных, а затем службу проверки, чтобы загрузить его из базы данных в механизм проверки.

Родительская тема: [Изучите тип SAP Commerce](#)

Предыдущий: [Пользовательские ограничения проверки](#)

Следующий: [Медиа-файлы](#)

## Напишите интеграционный тест для пользовательского ограничения

Проверьте, что ограничение работает должным образом, и внедрите службу проверки Hybris в код.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест Hybris123Tests.java:

```
public void testCustomConstraintIntegrationTest() {  
    assertTrue( checkTestSuiteXMLMatches("(.*).testsuite ошибки=\"0\" неудачи=\"0\" (.*) имя=\"NotLoremIpsumConstraintTest\" пакет=\"concerttours.cons\" )" );  
}
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testCustomConstraintIntegrationTest test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris.runtime.tests.Hybris123Tests#testCustomConstraintIntegrationTest test
```

## Процедура

. Загрузите ограничение через Impex, добавив новое ограничение Lorem Ipsum в essentialdata-ограничения.impexхле.

```
INSERT_UPDATE MinConstraint;id[unique=true];severity(code,itemtype(code));active;annotation;descriptor(enclosingType(code),qualifier);message[];AlbumSalesMustNotBeNegative;ERROR;Severity:true;javax.validation.constraints.Min;Band:albumSales;Albumverkäufe dürfen nicht negativ sein;Album
```

```
INSERT_UPDATE NotLoremIpsumConstraint;id[unique=true];severity(code,itemtype(code));active;annotation;descriptor(enclosingType(code),qualifier);NotLoremIpsum;ERROR;Severity:true;concerttours.constraints.NotLoremIpsum;Band:history;Band История не содержит Lorem ipsum Platzhalter Text.;Ba
```

Интерактивный ярлык SAP Commerce 123: если вы не можете добавить ограничение или хотите пропустить этот шаг, замените

<HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/impex/essentialdata-constraints.impex содержанием <HYBRIS\_HOME\_DIR>/

hybris123/src/main/webapp/resources/impex/validation/essentialdata-constraints-2.impex.

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/impex/validation/essentialdata-constraints-2.impex $HYBRIS_HOME_DIR/hybris/bin/custom/
```

```
echo a | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\impex\validation\essentialdata-constraints-2.impex %HYBRIS_HOME_DIR%\hybris
```

. Создайте интеграционный тест под названием NotLoremIpsumConstraintTest в концептные туры ограничения пакет под тесты сплаконцептные туры расширение.

```
упаковка concerttours.constraints; de.hybris.bootstrap.annotations.IntegrationTest;
```

```
импорт de.hybris.platform.servicelayer.ServicelayerTransactionalTest;
```

```
импорт de.hybris.platform.servicelayer.model.ModelService;
```

```
импорт de.hybris.platform.validation.exceptions.HybrisConstraintViolation;
```

```
импорт de.hybris.platform.validation.services.ValidationService; java.util.Set;
```

```
импорт
```

```
импорт
```

```
импорт javax.annotation.Resource;
```

```
импорт org.junit.Assert;
```

```
импорт org.junit.Do;
```

```
импорт org.junit.Test;
```

```
импорт модель BandModel концептных туров;
```

```
импорт модель NotLoremIpsumConstraintModel;
```

```
@Интеграционный тест
```

```
открытый класс NotLoremIpsumConstraintTest расширяет ServicelayerTransactionalTest {
```

```
    @Ресурс
```

```
    частная модель Сервис модель Сервис;
```

```
    @Ресурс
```

```
    частный Служба проверки Служба проверки;
```

```
    @До
```

```
    public void setup() выдает исключение {
```

```
        createCoreData();
```

```
        importCsv("/impex/essentialdata-constraints.impex", "UTF-8");
```

```
        validationService.reloadValidationEngine();
```

```
}
```

```
    @Тест
```

```
    public void testLoremIpsumConstraint() {
```

```
        окончательная BandModel band = modelService.create(BandModel.class);
```

```
        band.setCode("LoremIpsumTest1");
```

```
        band.setName("LoremIpsumBand");
```

```
        band.setHistory("Lorem Ipsum здесь");
```

```
        окончательный набор нарушений <HybrisConstraintViolation> = validationService.validate(band);
```

```
        Assert.assertTrue("Набор нарушений не должен быть нулевым или пустым", нарушения != null && нарушения.size() > 0); Assert.assertEquals("Должно быть одно нарушение ограничений", 1, нарушения.size());
```

```
        для (final HybrisConstraintViolation hybrisConstraintViolation : нарушения) {
```

```
            Assert.assertEquals(NotLoremIpsumConstraintModel.class, hybrisConstraintViolation.getConstraintModel().getClass());
```

```
}
```

```
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете добавить ограничение или хотите пропустить этот шаг, замените

<HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/testsrc/concerttours/constraints/NotLoremIpsumConstraintTest.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/

webapp/resources/concerttours/testsrc/concerttours/constraints/NotLoremIpsumConstraintTest.java.

```
то же самое $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/constraints/NotLoremIpsumConstraintTest.java $HYBR
```

```
echo f | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\testsrc\concerttours\constraints\NotLoremIpsumConstraintTest.
```

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Инициализируйте систему.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ant очистить все yunitinit
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и очистите все yunitinit
```

. Запустите интеграционные тесты.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ant integrationtests -Dtestclasses.packages="concerttours.*"
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и ant integrationtests -Dtestclasses.packages=concerttours.*
```

. Посмотреть результаты теста можно на <HYBRIS\_HOME\_DIR>/hybris/log/junit/index.html

. Запустите тесты CustomConstraintIntegrationTest повторите приемочное испытание и подтвердите, что оно теперь пройдено.

```
cd $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testCustomConstraintIntegrationTest test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testCustomConstraintIntegrationTest test
```

. Задокументируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR/hybris; git add .; git commit -m "Написать интеграционный тест для пользовательского ограничения"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Написать интеграционный тест для пользовательского ограничения"
```

## Медиа-файлы

SAP Commerce поддерживает файлы носителей. Файл носителя может быть чем угодно, что может быть сохранено в файловой системе.

Медиафайлы включают в себя файл Flash-анимации, изображение JPEG, видеофайл MPEG, файл CSV, текстовый файл, файл XML и многое другое. Платформа SAP Commerce не хранит медиафайлы в своей базе данных. Вместо этого она определяет тип элемента медиа, который называет и описывает конкретный файл медиа и знает, где файл физически хранится. Другими словами, элементы медиа в SAP Commerce — это не сами файлы медиа, а ссылки (URL) на эти файлы. Фактические файлы медиа, на которые есть ссылки, могут храниться в разных местах. Вы можете хранить их локально или хранить их удаленно, например, с помощью решений Amazon S3, Windows Azure Blob или MongoDB GridFS.

Помимо уникального идентификатора и описания, медиа-элемент имеет следующие атрибуты:

- Он назначен версии каталога. Медиа-элементы для продуктов могут быть синхронизированы между версиями каталога одновременно с соответствующими сведениями о продуктах.
- Доступ к нему можно контролировать с помощью списков разрешенных и запрещенных субъектов (пользователей).
- Он связан с форматом мультимедиа, который используется для маркировки различных элементов мультимедиа как подходящих для одних и тех же целей.
- Он включен в медиа-контейнер, обычно используемый для группировки медиа-файлов с одинаковым содержимым, представленных в разных форматах: например, одно и то же изображение, но разного размера или кодировки (png, jpg, bmp).

Чтобы сделать страницы групп немного интереснее, вы можете включить фотографии каждой группы. Используйте меньшую версию изображения для страницы списка и большую для страницы с подробностями.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Интеграционный тест для пользовательского ограничения](#)

Следующий:[Файлы свойств](#)

## Сопутствующая информация

[Управление цифровыми активами](#)

## Добавить медиафайлы

Добавьте изображения для группы uRock. Конечно, вы можете добавить изображения и для других.

## Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.java.

```
public void testBandImages() throws Exception {
    canLoginToHybrisCommerce();
    navigationTo("https://localhost:9002/concerttours/bands");
    waitForValidImage();

    navigationTo("https://localhost:9002/concerttours/bands/A006");
    assertTrue(waitForValidImage());
}
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
компакт-диск $HYBRIS_HOME_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testBandImages test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testBandImages test
```

## Процедура

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Изменить ГруппаБизнес-логика для включения изображений.

а. Добавьте атрибут контейнера мультимедиа в Группатип товара.

Добавьте следующее определение атрибута к типу элемента Band в concerttours-items.xml в папке ресурсов расширения concerttours:

```
<квалификатор атрибута="image" type="MediaContainer">
<description>изображение группы в разных форматах</description>
```

```
<настойчивость тип="свойство" />
</атрибут>
```

Интерактивный ярлык SAP Commerce 123: если вы не можете редактировать concerttours-items.xmlle, или вы хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttours-items.xmlc содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-itemsWithMedia.xml

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-itemsWithMedia.xml \$HYBRIS\_HOME\_DIR/hybris

echo a | xcopy %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\resources\concerttours-itemsWithMedia.xml %HYBRIS\_HOME\_D

6. Добавьте атрибут URL к BandData фасоль.

Добавьте следующее определение атрибута kBandData определение в concerttours-beans.xmlле в ресурсах папки концертные туры расширение:

```
<класс компонента = "concerttours.data.BandData" >
    <description>Объект данных, представляющий Band</description> <property>
        <имя свойства = "name" тип = "String" /> <имя свойства =
        <description> тип = "String" /> <имя свойства = "albumsSold" тип =
        "Long" />
        <имя свойства = "genres" тип="java.util.List<Строка>" />
        <имя свойства = "tours" тип="java.util.List<concerttours.data.TourSummaryData>" /> <имя свойства="imageURL"
        тип="String" />
    </бюб>
```

Интерактивный ярлык SAP Commerce 123: если вы не можете редактировать concerttours-beans.xmlle, или вы хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttours-beans.xmlc содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-beansWithMedia.xml

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-beansWithMedia.xml \$HYBRIS\_HOME\_DIR/hybris

echo a | xcopy %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\resources\concerttours-beansWithMedia.xml %HYBRIS\_HOME\_D

в. Обновите фасады, чтобы получить изображение в соответствующем формате.

Измените содержимое DefaultBandFacade класс для добавления обработки различных форматов изображений:

```
упаковка concerttours.facades.impl;
импорт de.hybris.platform.core.model.media.MediaContainerModel;
импорт de.hybris.platform.core.model.media.MediaFormatModel;
импорт de.hybris.platform.core.model.product.ProductModel;
импорт de.hybris.platform.servicelayer.media.MediaService; java.util.ArrayList;
импорт
импорт java.util.List;
импорт org.springframework.beans.factory.annotation.Required;
импорт
импорт concerttours.data.TourSummaryData;
импорт concerttours.enums.MusicType;
импорт concerttours.facades.BandFacade;
импорт concerttours.model.BandModel;
импорт concerttours.service.BandService;
импорт java.util.Locale;

открытый класс DefaultBandFacade реализует BandFacade {
    частный BandService bandService;
    частный МедиаСервис медиаСервис;
    @Переопределить
    публичный список<BandData> получитьBands() {
        окончательный список <BandModel> bandModels = bandService.getBands();
        окончательный список <BandData> bandFacadeData = new ArrayList<>();
        окончательный формат MediaFormatModel = mediaService.getFormat("bandList"); для
        (окончательный BandModel sm : bandModels) {
            окончательный BandData sfd = новый
            BandData(); sfd.setIid(sm.getCode());
            sfd.setName(sm.getName());
            sfd.setDescription(sm.getHistory(Locale.ENGLISH));
            sfd.setAlbumsSold(sm.getAlbumSales());
            sfd.setImageURL(getImageURL(sm, format));
            bandFacadeData.add(sfd);
        }
        вернуть bandFacadeData;
    }
    @Переопределить
    публичный BandData getBand(конечное имя строки) {
        если (имя == null) {
            throw new IllegalArgumentException("Название группы не может быть пустым");
        }
        окончательная BandModel band = bandService.getBandForCode(name); если
        (band == null)
        {
            вернуть ноль;
        }
        // Создать список жанров
        окончательный список<String> жанры = новый ArrayList<>();
        если (band.getTypes() != null)
        {
            для (финальный ТипМузыки musicType : band.getTypes()) {
                жанры.добавить(musicType.получитьКод());
            }
        }
        // Создаем список TourSummaryData из совпадений final
        List<TourSummaryData> tourHistory = new ArrayList<>(); if (band.getTours() != null)
        {
            для (финальный тип ProductModel: band.getTours()) {
                окончательная сводка TourSummaryData = new TourSummaryData();
```

```

summary.setId(tour.getCode());
summary.setTourName(tour.getName(Locale.ENGLISH));
// делаем большое предположение, что все варианты являются концертами и игнорируем каталоги продуктов
summary.setNumberOfConcerts(Integer.toString(tour.getVariants().size())); tourHistory.add(summary);

}

// Теперь мы можем создать объект передачи BandData
окончательный формат MediaFormatModel = mediaService.getFormat("bandDetail");
окончательный BandData BandData = новый BandData ();
BandData.setId(band.getCode()); BandData.setName(band.getName());
bandData.setAlbumsSold(band.getAlbumSales());
bandData.setImageURL(getImageURL(полоса, формат));
bandData.setDescription(band.getHistory(Locale.ENGLISH));
BandData.setGenres(жанры);

BandData.setTours(История тура);
вернуть данные полосы;
}

защитенная строка getImageURL(окончательный BandModel sm, окончательный формат MediaFormatModel) {

окончательный контейнер MediaContainerModel = sm.getImage();
если (контейнер != null)
{
    return mediaService.getMediaByFormat(контейнер, формат).getDownloadURL();
}
вернуть ноль;
}

@Необходимый
public void setBandService(final BandService bandService) {

этот.bandService = bandService;
}

@Необходимый
public void setMediaService(final MediaService mediaService) {

этот.mediaService = mediaService;
}
}

```

-Примечание

С помощью этой модификации вы делаете следующее:

- Вы используете mediaService для получения конкретных моделей форматов носителей.
- В получитьBands() метод, вы извлекаете bandList формат изображения, уменьшенная версия изображения.
- В получитьBand() метод, мы извлекаем bandDetail формат изображения, увеличенная версия изображения.
- Вы возвращаете URL-адрес изображения, которое необходимо отобразить.
- Вы жестко закодировали зависимость от наличия двух определенных форматов медиа. Если их нет, выдается исключение. Два формата медиа стали существенными элементами данных.

Необязательное улучшение, которое вы могли бы сделать, — это возврат изображения по умолчанию, когда Band еще не предоставил изображение. Другое улучшение также вернуло бы любой alt-text, указанный для изображения в объекте передачи данных (DTO).

Интерактивный ярлык SAP Commerce 123: если вы не можете редактировать DefaultBandFacade, или вы хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/facades/impl/DefaultBandFacade.java с содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/facades/impl/DefaultBandFacadeWithMedia.java.

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/facades/impl/DefaultBandFacadeWithMedia.java \$HY

echo a | xcopy %HYBRIS\_HOME\_DIR%hybris123srcmainwebappresourcesconcerttourssrcconcerttoursfacadesimplDefaultBandFacadeWithMedia

г. Обновление концертные туры-весна.xml для инъекции DefaultBandFacade с Медиа Сервисом.

```

<имя псевдонима = "defaultBandFacade" псевдоним = "bandFacade" />
<bean id = "defaultBandFacade" class = "concerttours.facades.impl.DefaultBandFacade" >
    <имя свойства = "bandService" ref = "bandService" /> <имя
    свойства="mediaService" ref="mediaService"/> </bean>

```

Интерактивный ярлык SAP Commerce 123: если вы не можете редактировать concerttours-beans.xml, или вы хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttours-spring.xml с содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-spring-withMedia.xml

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-spring-withMedia.xml \$HYBRIS\_HOME\_DIR/hybris

echo a | xcopy %HYBRIS\_HOME\_DIR%hybris123srcmainwebappresourcesconcerttoursresourcesconcerttours-spring-withMedia.xml %HYBRIS\_HOME

. Инициализируйте и обновите вашу систему.

cd \$HYBRIS\_HOME\_DIR/hybris/bin/platform; ant clean all build initialize updatesystem

cd %HYBRIS\_HOME\_DIR%hybris\bin\platform и ant clean all build initialize updatesystem

. Измените Band JSP, включив в них изображения, обновив BandList.jsp и BandDetails.jsp страницы следующим образом.

```

<%@taglib  префикс="c"  uri="http://java.sun.com/jsp/jstl/core"%>
<! тип документа html>
<html>
<title>Группа Подробности</title>
<тело>
    <h1>Группа Подробности</h1>
    Подробная информация о группе ${band.name}
    <p></p> <p>$
    {band.description}</p>
    <p>Тип музыки:</p>
    <ул>

```

```

<c:forEach var="жанр"           элементы="${band.genres}">
    <li>${жанр}</li>
</c:forEach>
</ul>
<p>История тура:</p>
<ul>
    <c:forEach var="тур"           элементы="${band.tours}">
        <li><a href="#">${тур.tourName}</a>(${тур.numberOfConcerts})</li> </c:forEach>
    </ul>
    <a href="#">Вернуться к списку групп</a> </body>
</html>

```

```

<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %> <!doctype html>

<html>
    <title>Группа Список</title>
    <тело>
        <h1>Группа Список</h1>
        <уна>
            <c:forEach var="band" items="${bands}">
                <li><a href="#">${band.id}</a> ${band.name}</a></li> </c:forEach>
            </ul>
        </тело>
    </html>

```

SAP Commerce 123 Интерактивный ярлык: Если вы не можете редактировать файлы JSP или хотите пропустить этот шаг, замените кнопку, чтобы создать два новых формата медиа. Первый с квалификацией <*HYBRIS\_HOME\_DIR*>/hybris/bin/custom/concerttours/web/webroot/WEB-INF/views/BandList.jspc содержанием <*HYBRIS\_HOME\_DIR*>/hybris123/src/main/webapp/resources/concerttours/web/webroot/WEB-INF/views/BandListWithMedia.jsp, и <*HYBRIS\_HOME\_DIR*>/hybris/bin/custom/concerttours/web/webroot/WEB-INF/views/BandDetails.jspc содержанием <*HYBRIS\_HOME\_DIR*>/hybris123/src/main/webapp/resources/concerttours/web/webroot/WEB-INF/views/BandDetailsWithMedia.jsp.

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/web/webroot/WEB-INF/views/BandListWithMedia.jsp \$HYBRIS\_HOME\_DIR/hybris

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/web/webroot/WEB-INF/views/BandDetailsWithMedia.jsp \$HYBRIS\_HOME\_DIR/hybris

echo a | xcopy %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\web\webroot\WEB-INF\views\BandListWithMedia.jsp %HYBRIS\_HOME\_

echo a | xcopy %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\web\webroot\WEB-INF\views\BandDetailsWithMedia.jsp %HYBRIS\_HO

.Запустите SAP Commerce.

cd \$HYBRIS\_HOME\_DIR/hybris/bin/platform; ./hybrisserver.sh start

cd %HYBRIS\_HOME\_DIR%\hybris\bin\platform и запустить hybrisserver.bat

.Добавить два новых медиаформата.

а. В административной панели бэк-офиса перейдите к **Форматы мультимедиа** элемент в дереве проводника.

б. Нажмите **наплоскнопка** для создания двух новых медиаформатов. Первый с квалификацией **bandDetail** и имя **формат детализации полосы**, и второй с квалификацией **bandList** и имя **формат списка групп**.

.Найдите соответствующие изображения для каждой из шести групп.

.Используйте инструмент редактирования изображений, чтобы создать две версии JPEG каждого изображения: одну шириной 600 пикселей для страницы сведений и одну шириной 100 пикселей для страницы списка.

Дайте изображениям следующие названия:

- bigbandBig.jpg
- bigbandSmall.jpg
- danceBig.jpg
- danceSmall.jpg
- jazzBig.jpg
- jazzSmall.jpg
- омфаBig.jpg
- омфаSmall.jpg
- оркестрБольшой.jpg
- orchestraSmall.jpg
- rockBig.jpg
- rockSmall.jpg

.Добавьте два новых медиа-элемента для группы yRock.

а. В административной панели бэк-офиса перейдите к **Мультиформаты Медиа** элемент в дереве проводника.

б) Создайте один новый медиа-элемент с идентификатором **RockSmall** и каталогная версия **ConcerttoursKatalog** продукции: Интернет.

с. Добавить второй новый медиа-элемент с идентификатором **RockLarge** и каталогная версия **ConcerttoursKatalog** продукции: Интернет.

.Создайте медиа-контейнер для изображений группы yRock.

а. В административной панели бэк-офиса перейдите к **Мультиформаты Медиа** элемент в дереве проводника.

б. Нажмите **наплоскнопка** для создания нового медиаконтейнера с именем и квалификацией, установленными на **RockImageContainer**.

.Добавьте изображение к каждому элементу.

а. В административной панели бэк-офиса перейдите к **Мультиформаты Медиа** элемент в дереве проводника.

б. Щелкните **поиск** для перечисления всех элементов мультимедиа, включая **rockSmall.jpg** и **rockLarge.jpg**.

в. Нажмите **hayRockSmall**, загрузить изображение **rockSmall.jpg**, указать **bandList** как формат СМИ, и **RockImageContainer** как Медиа-контейнер.

г. Нажмите на **RockLarge**, загрузить изображение **rockBig.jpg** и укажите **bandDetail** как формат СМИ, и **RockImageContainer** как Медиа-контейнер.

. Назначьте контейнер мультимедиа соответствующей полосе.

а. В административной панели бэк-офиса перейдите к **Концертные туры** в панели проводника.

б) Найдите группу **yRock**.

с. Установите его **BandImage** поле **kyRockImageContainer**.

. Убедитесь, что вы видите изображения на страницах групп.

Вы должны увидеть изображения на следующих страницах:

- <https://localhost:9002/concerttours/bands>
- <https://localhost:9002/concerttours/bands/A001>

. Чтобы определить несколько элементов мультимедиа, форматов и контейнеров, используйте ImpEx и определите их как основные данные, чтобы элементы формата мультимедиа загружались в базу данных до загрузки данных проекта.

а. Создайте новый файл с именем **essentialdata-mediaformats.impex** в **расширение концертных туров** и установив содержимое следующим образом.

```
ВСТАВИТЬ_ОБНОВЛЕНИЕ MediaFormat;qualifier[unique=true];name Формат
;список_групп;группа списка
;bandDetail;Полоса Формат деталей
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать файл ImpEx или хотите пропустить этот шаг, замените **<HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/impex/media/essentialdata-mediaformats.impex** содержанием **<HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/impex/**

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/impex/media/essentialdata-mediaformats.impex $HYBRIS_HOME_DIR/hybris/bin/custom/co
```

```
копировать %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\impex\media\essentialdata-mediaformats.impex %HYBRIS_HOME_DIR%\hybris\bin\custo
```

б) Добавьте медиа-файлы в проект, скопировав изображения групп в новую папку с именем **bandimages** рамках существующих ресурсов/концертных туров в концертную папку в концертные туры расширение.

Интерактивный ярлык SAP Commerce 123: если вы не можете добавить файлы мультимедиа или хотите пропустить этот шаг, замените **<HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/bandimages/\*** содержанием **<HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttours/bandimages/**

```
mkdir -p $HYBRIS_HOME_DIR/hybris/bin/custom/concerttours/resources/concerttours/bandimages; cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/
```

```
эхо | xcopy %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\bandimages\* %HYBRIS_HOME_DIR%\hybris\bin\custom\concert
```

с. Добавьте медиа-данные в файлы ImpEx полос, обновив содержимое концертных туров-группы. **импех** в **ресурсах/импех** концертных туров расширение следующим образом.

```
# ImpEx для импорта групп в магазин Little Concert Tours
```

```
# Макросы / Определения параметров замены $productCatalog=concerttoursProductCatalog $catalogVersion=catalogversion(catalog(id[default=$productCatalog]),version[default='Online']) [unique=true,default=$productCatalog:$medias=medias(code, $catalogVersion)]
```

```
$siteResource=jar:concerttours.constants.ConcerttoursКонстанты&/concerttours/bandimages
```

```
# Каталог продукции
INSERT_UPDATE Каталог;id[unique=true];
$productCatalog
```

```
# Версия каталога продукции
INSERT_UPDATE CatalogVersion;catalog(id[unique=true];version[unique=true];active;languages(isoCode);readPrincipals(uid));
$productCatalog;Online;true;en;employeegroup
```

```
INSERT_UPDATE Медиа;mediaFormat(квалификатор);code[unique=true];@media[translator=de.hybris.platform.impex.jalo.media.MediaDataTranslator];mi ;bandList;rockSmall.jpg;
$siteResource/rockSmall.jpg;;
;bandDetail;rockBig.jpg;$siteResource/rockBig.jpg;;bandList;danceSmall.jpg;$siteResource/
danceSmall.jpg;;bandDetail;danceBig.jpg;$siteResource/danceBig.jpg;;bandList;jazzSmall.jpg;
$siteResource/jazzSmall.jpg;;bandDetail;jazzBig.jpg;$siteResource/
jazzBig.jpg;;bandList;bigbandSmall.jpg;$siteResource/bigbandSmall.jpg;;bandDetail;bigbandBig.jpg;
$siteResource/bigbandBig.jpg;;bandList;omphaSmall.jpg;$siteResource/
omphaSmall.jpg;;bandDetail;omphaBig.jpg;$siteResource/omphaBig.jpg;;bandList;orchestraSmall.jpg;
$siteResource/orchestraSmall.jpg;;bandDetail;orchestraBig.jpg;$siteResource/orchestraBig.jpg;;
INSERT_UPDATE MediaContainer;квалификатор[ unique=true];$medias;
$catalogVersion ;yRockImage;rockSmall.jpg,rockBig.jpg;
```

```
;yBandImage;danceSmall.jpg,danceBig.jpg; ;yJazzImage;jazzSmall.jpg,jazzBig.jpg; ;BannedImage;bigbandSmall.jpg,bigbandBig.jpg; ;SirkenImage;omphaSmall.jpg,omphaBig.jpg; ;TheChoirImage;orchestra
```

```
INSERT_UPDATE
```

```
Группа;код[уникальный=истина];название;альбомПродажи;изображение(квалификатор) ;A001;yRock;1000000;yRockImage
```

```
;A006;yBand;yBandImage ;A003;yJazz;7;yJazzImage ;A004;Banned;427;BannedImage ;A002;Sirken;2000;SirkenImage ;A005;The
```

```
Xop;49000;TheChoirImage
;A007; Тихо;1200;;
```

-Примечание

Одна полоса остается без связанных изображений, чтобы проверить, что система правильно обрабатывает эту ситуацию.

Интерактивный ярлык SAP Commerce 123: если вы не можете создать файл ImpEx или хотите пропустить этот шаг, замените **<HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/impex/media/concerttours-bands.impex** содержанием **<HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/impex/**

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/impex/media/concerttours-bands.impex $HYBRIS_HOME_DIR/hybris/bin/custom/concerttou
```

копировать %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\impex\media\concerttours-bands.impex %HYBRIS\_HOME\_DIR%\hybris\bin\custom\conce

г. Остановить SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

е. Инициализируйте систему.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; инициализация ant
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и инициализация ant
```

. Запустите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh start
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и запустить hybrisserver.bat
```

. Убедитесь, что вы видите изображения на страницах группы.

Вы должны увидеть изображения на следующих страницах:

- <https://localhost:9002/concerttours/bands> https://localhost:9002/concerttours/bands/A001 . Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR/hybris/bin/platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Исправьте тестовые случаи.

а. Добавить звонок в импорт Csv настраивать() Метод DefaultBandFacadeIntegrationTest.

```
@Do
public void setUp() выдает исключение {
    try {
        Thread.sleep(TimeUnit.SECONDS.toMillis(1));
        new JdbcTemplate(Registry.getCurrentTenant().getDataSource()).execute("CHECKPOINT");
        Thread.sleep(TimeUnit.SECONDS.toMillis(1));
    } catch (InterruptedException exc) {} importCsv("/impex/essentialdata-mediaformats.impex", "UTF-8");
    // Этот экземпляр BandModel будет использоваться тестами bandModel =
    modelService.create(BandModel.class); bandModel.setCode(BAND_CODE);
    bandModel.setName(Имя_ДИАПАЗОНА);
    bandModel.setHistory(BAND_HISTORY);
    bandModel.setAlbumSales(ALBUMS SOLD);
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете редактировать интеграционный тест или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/testsrc/concerttours/facades/impl/DefaultBandFacadeIntegrationTest.javac содержанием

```
<HYBRIS_HOME_DIR>/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/facades/impl/DefaultBandFacadeIntegrationT
```

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/facades/impl/DefaultBandFacadeIntegrationTestWi
```

копировать /у %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\testsrc\concerttours\facades\impl\DefaultBandFacadeIntegrationT

б. Добавить звонок импорт Csv testBandServiceTours() Метод Тест интеграции службы Band по умолчанию sort.

```
@Test
public void testBandServiceTours() выдает исключение {
    createCoreData();
    importCsv("/impex/essentialdata-mediaformats.impex", importCsv("/impex/UTF-8");
    concerttours-bands.impex", "utf-8"); importCsv("/impex/concerttours-
    yBandTour.impex", "utf-8"); final BandModel band =
    bandService.getBandForCode("A001"); assertNotNull("Группа не найдена",
    band);
    final Set<ProductModel> tours = band.getTours();
    assertNotNull("Тур не найден", tours);
    Assert.assertEquals("не найдено ни одного тура", 1, tours.size()); final Object[]
    objects = new Object[5];
    final Collection<VariantProductModel> concerts = ((ProductModel) tours.toArray(objects)[0]).getVariants(); assertNotNull("Тур не найден", tours);
    Assert.assertEquals("не найдено ни одного тура", 6, concerts.size());
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете редактировать интеграционный тест или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/testsrc/concerttours/service/impl/DefaultBandServiceIntegrationTest.javac содержанием

```
<HYBRIS_HOME_DIR>/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/service/impl/DefaultBandServiceIntegration
```

```
cp $HYBRIS_HOME_DIR/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/service/impl/DefaultBandServiceIntegrationTestWi
```

```
копия /г %HYBRIS_HOME_DIR%\hybris123\src\main\webapp\resources\concerttours\tests\concerttours\service\impl\DefaultBandServiceIntegration
```

в) Инициализируйте систему.

```
cd $HYBRIS_HOME_DIR\hybris\bin\platform; ant очистить все yunitinit
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и очистите все yunitinit
```

г. Запустите тесты, чтобы убедиться, что все они пройдены.

```
hybris всетесты -Dtestclasses.packages="concerttours.*"
```

```
hybris всетесты -Dtestclasses.packages=концертные туры.*
```

. Запустите SAP Commerce.

```
cd $HYBRIS_HOME_DIR\hybris\bin\platform; ./hybrisserver.sh start
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и запустить hybrisserver.bat
```

. Запустите testBandImagesПовторите приемочное испытание и подтвердите, что оно прошло успешно.

```
компакт-диск $HYBRIS_HOME_DIR\hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testBandImages тест
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testBandImages test
```

. Зафиксируйте изменения в локальном репозитории Git.

```
cd $HYBRIS_HOME_DIR\hybris; git add . ; git commit -m "Добавить медиафайлы"
```

```
cd %HYBRIS_HOME_DIR%\hybris & git add . & git commit -m "Добавить медиафайлы"
```

## Файлы свойств

SAP Commerce опирается на два основных файла конфигурации: project.properties и local.properties. Свойства проекта являются значениями по умолчанию SAP Commerce, в то время как локальные свойства — это то место, где вы можете определить собственную конфигурацию для своего расширения.

Вы можете задать значения в различных файлах свойств. Каждый файл имеет свой приоритет, и его значения могут переопределять значения другого файла с более низким приоритетом. Порядок приоритета от высокого к низкому:

.местные.свойстваale — это рабочая копия свойства проекта, расположенный в <HYBRIS\_CONFIG\_DIR>каталог. Позволяет переопределить настройки по умолчанию из свойства проекта. Используйте местные.свойства для установки значений свойств конфигурации, которые необходимо настроить для вашего проекта.

.Специфическое расширение свойства проекта находится в <HYBRIS\_BIN\_DIR>/<EXTENSION\_DIR>. Он определяет значения, используемые для расширения.

Глобальный свойства проекта находится в <HYBRIS\_BIN\_DIR>/платформа каталог, и предоставляет заводские настройки по умолчанию. Не рекомендуется редактировать этот файл.

Родительская тема:[Изучите тип SAP Commerce](#)

Предыдущий:[Медиа-файлы](#)

Сопутствующая информация

[Настройка поведения SAP Commerce](#)

### Определить значения свойств по умолчанию

Создать свойства проекта файл, который определяет значения свойств конфигурации по умолчанию для вашего расширения.

### Предпосылки

В ваш комплект документов включен следующий приемочный тест.Hybris123Tests.java.

```
public void testPropertiesFiles() { assertTrue(
checkTestSuiteXMLMatches("(.)testsuite ошибки=\"0\" неудачи=\"0\" (.*) имя=\"DefaultBandFacadeIntegrationWithPropertiesTest\" пакет=\"concerttou {
```

Прежде чем приступить к этому шагу, запустите тест, чтобы убедиться в его неудаче, выполнив следующую команду:

```
cd $HYBRIS_HOME_DIR\hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testPropertiesFiles test
```

```
cd %HYBRIS_HOME_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testPropertiesFiles test
```

### Процедура

. Остановите SAP Commerce.

```
cd $HYBRIS_HOME_DIR\hybris\bin\platform; ./hybrisserver.sh остановить
```

```
cd %HYBRIS_HOME_DIR%\hybris\bin\platform и hybrisserver.bat остановить
```

. Обновите класс фасада для использования значений свойств.

Редактировать DefaultBandFacade класс в концептные туры.фасады.импл пакет подисточник каталог концептные туры расширение. Замените жестко заданные значения на имена свойств следующим образом:

```
упаковка concerttours.facades.impl;
импорт de.hybris.platform.core.model.media.MediaContainerModel;
импорт de.hybris.platform.core.model.media.MediaFormatModel;
импорт de.hybris.platform.core.model.product.ProductModel;
импорт de.hybris.platform.servicelayer.config.ConfigurationService;
импорт de.hybris.platform.servicelayer.media.MediaService; java.util.ArrayList;
импорт
импорт java.util.List;
импорт org.springframework.beans.factory.annotation.Required;
импорт
импорт concerttours.data.TourSummaryData;
импорт concerttours.enums.MusicType;
импорт concerttours.facades.BandFacade;
импорт concerttours.model.BandModel;
импорт concerttours.service.BandService;
импорт java.util.Locale;

открытый класс DefaultBandFacade реализует BandFacade {
    публичная статическая окончательная строка BAND_LIST_FORMAT = "band.list.format.name";
    частная статическая окончательная строка BAND_DETAIL_FORMAT = "band.detail.format.name";
    частная BandService bandService;
    частный МедиаСервис медиа Сервис;
    частный КонфигурацияСервис configService;
    @Переопределить
    публичный список<BandData> получитьBands()
    {
        окончательный список<BandModel> bandModels = bandService.getBands();
        окончательный список<BandData> bandFacadeData = new ArrayList<>(); если
        (bandModels!=null && !bandModels.isEmpty()) //6.2 {
            окончательная строка mediaFormatName = configService.getConfiguration().getString(BAND_LIST_FORMAT);
            System.out.println("mediaFormatName:" + mediaFormatName);
            окончательный формат MediaFormatModel = mediaService.getFormat(mediaFormatName); для
            (окончательный BandModel sm : bandModels)
            {
                окончательный BandData sfd = новый
                    BandData(); sfd.setId(sm.getCode());
                sfd.setName(sm.getName());
                sfd.setDescription(sm.getHistory(Locale.ENGLISH));
                sfd.setAlbumsSold(sm.getAlbumSales());
                sfd.setImageURL(getImageURL(sm, format));
                bandFacadeData.add(sfd);
            }
        }
        вернуть bandFacadeData;
    }

    @Переопределить
    public BandData getBand(конечное имя строки) {
        если (имя == null) {
            throw new IllegalArgumentException("Название группы не может быть пустым");
        }
        окончательная BandModel band = bandService.getBandForCode(name); если
        (band == null)
        {
            вернуть ноль;
        }
        // Создать список жанров
        окончательный список<String> жанры = новый ArrayList<>();
        если (band.getTypes() != null)
        {
            для (финальный ТипМузыки musicType : band.getTypes()) {
                жанры.добавить(musicType.получитьКод());
            }
        }
        // Создать список TourSummaryData
        окончательный список<TourSummaryData> tourHistory = new ArrayList<>(); если
        (band.getTours() != null)
        {
            для (финальный тип ProductModel: band.getTours()) {
                окончательная сводка TourSummaryData = new TourSummaryData();
                summary.setId(tour.getCode());
                summary.setTourName(tour.getName(Locale.ENGLISH));
                // делаем большое предположение, что все варианты являются концертами и игнорируем каталоги продуктов
                summary.setNumberOfConcerts(Integer.toString(tour.getVariants().size())); tourHistory.add(summary);
            }
        }
        // Теперь мы можем создать объект передачи BandData
        окончательный формат MediaFormatModel = mediaService.getFormat(mediaFormatName);

        окончательный BandData BandData = новый BandData
        (); BandData.setId(band.getCode());
        bandData.setName(band.getName());
        bandData.setAlbumsSold(band.getAlbumSales());
        bandData.setImageURL(getImageURL(band, format));
        bandData.setDescription(band.getHistory(Locale.ENGLISH));
        bandData.setGenres(genres);
        BandData.setTours(История тура);
        вернуть данные полосы;
    }

    защищенная строка getImageURL(окончательный BandModel sm, окончательный формат MediaFormatModel) {
        окончательный контейнер MediaContainerModel = sm.getImage();
        если (контейнер != null)
        {
            return mediaService.getMediaByFormat(контейнер, формат).getDownloadURL();
        }
    }
}
```

```

        }
        возвращаться нулевой;
    }

    @Необходимый
    public void setBandService(final BandService bandService) {
        этот.bandService = bandService;
    }

    @Необходимый
    public void setMediaService(final MediaService mediaService) {
        этот.mediaService = mediaService;
    }

    @Необходимый
    public void setConfigurationService(final ConfigurationService configService) {
        этот.configService = configService;
    }
}

```

Интерактивный ярлык SAP Commerce 123: если вы не можете обновить фасад самостоятельно или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/src/concerttours/facades/impl/DefaultBandFacade.javac содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/src/concerttours/facades/impl/DefaultBandFacadeWithProperties.java

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/src/concerttours/facades/impl/DefaultBandFacadeWithProperties.java \$HYB

echo a | xcopy %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\src\concerttours\facades\impl\DefaultBandFacadeWithProperties

. Обновите конфигурацию Spring, добавив инъекцию Конфигурация Сервисов концертные туры-весна.xml в ле.

Конфигурация для defaultBandFacade теперь выглядит следующим образом:

```

<имя псевдонима = "defaultBandFacade" псевдоним = "bandFacade" />
<bean id = "defaultBandFacade" class = "concerttours.facades.impl.DefaultBandFacade" >
    <имя свойства = "bandService" ref = "bandService" /> <имя
    свойства="mediaService" ref="mediaService"/>
    <имя_свойства="configurationService" ref="configurationService" /> </bean>

```

Интерактивный ярлык SAP Commerce 123: если вы не можете обновить Spring XML самостоятельно или хотите пропустить этот шаг, замените <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/resources/concerttours-spring.xmlc содержанием <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-spring-withConfigurationService.xml

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/resources/concerttours-spring-withConfigurationService.xml \$HYBRIS\_HOME

echo a | xcopy %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\resources\concerttours-spring-withConfigurationService.xml %H

. Определить свойства проекта по умолчанию.

Добавьте следующие строки в свой концертные туры расширения свойства проекта ле.

```

# имена форматов медиа
band.list.format.name =
band.detail.format.name = bandDetail

```

Интерактивный ярлык SAP Commerce 123: если вы не можете выполнить обновление свойства проекта или вы хотите пропустить этот шаг, добавьте содержимое <HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/projectpropertieswithbands.txtk <HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/project.properties

cp \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/projectpropertieswithbands.txt \$HYBRIS\_HOME\_DIR/hybris/bin/custom/concertt

копировать %HYBRIS\_HOME\_DIR%\hybris123\src\main\webapp\resources\concerttours\projectpropertieswithbands.txt %HYBRIS\_HOME\_DIR%\hybris\bin\custom\conc

. Создайте новый интеграционный тест для проверки DefaultBandFacade фасад при использовании этих новых свойств путем создания тестового класса с именем concerttours.facades.impl.DefaultBandFacadeIntegrationWithPropertiesTest под тесты гспака концертные туры расширение.

```

упаковка concerttours.facades.impl; static
импорт org.junit.Assert.assertTrue;
импорт java.lang.InterruptedException; java.util.List;
импорт
импорт java.util.concurrent.TimeUnit;

импорт de.hybris.bootstrap.annotations.IntegrationTest;
импорт de.hybris.platform.servicelayer.ServicelayerTransactionalTest;
импорт javax.annotation.Resource;

импорт org.junit.After;
импорт org.junit.Do;
импорт org.junit.Test;
импорт concerttours.data.BandData;
импорт concerttours.facades.BandFacade;
импорт de.hybris.platform.core.Registry;
импорт org.springframework.jdbc.core.JdbcTemplate;

```

@Интеграционный тест  
открытый класс DefaultBandFacadeIntegrationWithPropertiesTest расширяет ServicelayerTransactionalTest {

@Ресурс  
частный BandFacade bandFacade;

@До  
public void setUp() выдает исключение {

```

    пытаться (
        Thread.sleep(TimeUnit.SECONDS.toMillis(1));
        new JdbcTemplate(Registry.getCurrentTenant().getDataSource()).execute("CHECKPOINT");
        Thread.sleep(TimeUnit.SECONDS.toMillis(1));
    )

```

12/3/24, 11:01 утра

```
        }
        поймать (InterruptedException exc) {
    }

    @Test
    public void testProperties() выдает исключение {

        createCoreData();
        importCsv("/impeкс/essentialdata-mediaformats.impeкс", importCsv("/импекс/withLocalization/tours-
bands.impeкс", "UTF-8"); importCsv("/импекс/withLocalization/concerttours-
yBandTour.impe克斯", "UTF-8"));

        Список<BandData> полосы = bandFacade.getBands();
        assertTrue(bands.size() > 0);
        assertEquals( DefaultBandFacade.BAND_LIST_FORMAT.equals("band.list.format.name"));

    }

    @После
    public void teardown() {

    }
}
```

Интерактивный ярлык SAP Commerce 123: если вы не можете создать тестовый класс самостоятельно или хотите пропустить этот шаг, замените  
<HYBRIS\_HOME\_DIR>/hybris/bin/custom/concerttours/testsrc/concerttours/facades/impl/DefaultBandFacadeIntegrationWithPropertiesTest.java с содержанием

<HYBRIS\_HOME\_DIR>/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/facades/impl/DefaultBandFacadeIntegrationWithPr

то же самое \$HYBRIS\_HOME\_DIR/hybris123/src/main/webapp/resources/concerttours/testsrc/concerttours/facades/impl/DefaultBandFacadeIntegrationWithPrope

echo F | xcopy %HYBRIS\_HOME\_DIR%hybris123\src\main\webapp\resources\concerttours\testsrc\concerttours\facades\impl\DefaultBandFacadeIntegratio

. Перестройте SAP Commerce с помощью Ant.

cd \$HYBRIS\_HOME\_DIR/hybris/bin/platform; ant очистить все

cd %HYBRIS\_HOME\_DIR%\hybris\bin\platform и очистите все

. Запустите интеграционные тесты.

cd \$HYBRIS\_HOME\_DIR/hybris/bin/platform; ant integrationtests -Dtestclasses.packages="concerttours.\*"

cd %HYBRIS\_HOME\_DIR%\hybris\bin\platform и ant integrationtests -Dtestclasses.packages=concerttours.\*

. Посмотреть результаты теста можно на <HYBRIS\_HOME\_DIR>/hybris/log/junit/index.html подтверждаю, что DefaultBandFacadeИнтеграцияСоСвойствамиТест прошло.

. Запустите testPropertiesFiles повторите приемочное испытание и подтвердите, что оно теперь пройдено.

cd \$HYBRIS\_HOME\_DIR/hybris123; mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testPropertiesFiles test

cd %HYBRIS\_HOME\_DIR%\hybris123 & mvn -Dtest=com.hybris.hybris123.runtime.tests.Hybris123Tests#testPropertiesFiles test

. Зафиксируйте изменения в локальном репозитории Git.

cd \$HYBRIS\_HOME\_DIR/hybris; git add . ; git commit -m "Определить значения свойств по умолчанию"

cd %HYBRIS\_HOME\_DIR%\hybris & git add . & git commit -m "Определить значения свойств по умолчанию"

. Очистить <INITIAL\_ADMIN> переменная окружения.

а. Откройте профиль оболочки с помощью текстового редактора командной строки.

пико ~/.bash\_profile

б) Замените содержимое файла следующими строками.

```
экспорт HYBRIS_HOME_DIR=/opt/CXCOMM220500P_X-XXXXXXX ANT_HOME=$
экспорт {HYBRIS_HOME_DIR}/hybris/bin/platform/apache-ant ПУТЬ=${PATH}:$#
экспорт {ANT_HOME}/bin
```

setx INITIAL\_ADMIN "" /m

## Результаты

Поздравляем! Вы завершили тур. Теперь вы должны хорошо понимать основные концепции SAP Commerce.

## Следующие шаги

Если вы проходите несколько экскурсий, удалите все разархивированные папки SAP Commerce и перезагрузите компьютер между экскурсиями, чтобы убедиться, что нет запущенных потоков SAP Commerce. Затем выполните шаги в [Прежде чем начать](#) и начните свой следующий тур.