Промежуточные результаты компиляции проекта game-server под разными версиями джавы.

// java 8 ============================================

Чтобы скомпилировать проект под джава8, пришлось в POM отключить некоторые плагины, которые что-то копируют, перемещают, удаляют. Все сторонние библиотеки, используемые в проекте, фактически уже сильно deprecated. Чтобы проект собрался под джавой8 пришлось изменить некоторые версии библиотек на более свежие. Например, для jdbc драйвера mssqlserver версия 4 была изменена на версию 8.2

Фактически, весь проект написан как большой монолит под джава 6, что указано в мавене (pom.xml):

<configuration>

<source>1.6</source>

<target>1.6</target>

<encoding>iso-8859-1</encoding>

</configuration>

В итоге проект собрался 8й джавой с кучей предупреждений про deprecated код. (см. game-server-compile-j8.txt).

Изменение версии спринга с 2.5.6 на 5.2.7 вызвало многочисленные ошибки компилятора (game-server-compile-spring-5-2-7-j8.txt). Некоторые пакеты в спринге 5.2.7 удалены или перемещены физически, а именно:

- org.springframework.jdbc.object

- org.springframework.context

- org.springframework.orm.hibernate3.support

- org.springframework.jdbc.core

405 директив импорта из этих пакетов.

Большая зависимость в проекте спринга 2.5.6 от класса - org.springframework.orm.hibernate3.support.HibernateDaoSupport (99 наследников); Этот класс завязан на хибернейт3. В пятом спринге уже используется хибернейт5:

org.springframework.orm.hibernate5.support.HibernateDaoSupport

Адаптировать весь проект под спринг5\хибернейт5 – неизвестно сколько времени займет, и не факт, что не упремся в новые подводные камни.

Если разработчики premiumChess не мигрировали в свое время на джаву8\спринг5, то, видимо, этому тоже были причины. Под спринг 5.2.7 надо рефакторить весь проект, и не факт что взлетит после первого билда.

// java 14 ===========================================

Собрать проект под джава-14 так и не удалось (game-server-compile-j14.txt). Основная проблема, что джава 14 уже не хочет поддерживать и компилить исходники под джава 6.

Source option 6 is no longer supported. Use 7 or later.

Target option 6 is no longer supported. Use 7 or later.

Изменение директив source и target с версии 1.6 на 1.7 также не помогло – в 14й джаве некоторые пакеты также удалены физически (game-server-compile-source7-j14.txt)

-package javax.jws does not exist.

Это основной пакет, который использовался в проекте для создания и объявления вэбсервисов. Под джава 14 потребуется также переделки всего проекта. Программирование с пакетами этого класса довольно громоздкое.

// core-server-1.0.1.jar ====================================

Выяснился ещё такой ньюанс с этой библиотекой. У нас нет её исходников, только сама либа с откомпиленными клаcсами\интерфейсами:

* CoreGame
* CoreMove
* CoreGameDescription
* CoreChallengeGameColors
* CriticalPlayer
* ChallengeColorSelectServiceImpl
* ServerMoveValidationService
* ChalengeColorSelectService
* GameData
* И ещё 16 классов

Собрана она под джава6 в 2012г одним из разработчиков, чей никнейм часто встречается в коде в качестве автора - melchiori:

Manifest-Version: 1.0

Archiver-Version: Plexus Archiver

Created-By: Apache Maven

Built-By: melchiori

Build-Jdk: 1.6.0\_17

Ньюанс в том, что в коде часто ссылаются на классы\интерфейсы этой либы при объявлении новых классов, без неё проект не собирается, а её исходников у нас нет. Например,

public class Player extends Cacheable implements ClusteredEntity, CriticalPlayer

// Общее резюме ======================================

Миграция и адаптация всего проекта под джава8\14 + спринг 5.2 + хибернейт 5 – неизвестно сколько времени это займет, и неизвестно, какие рантайм-ошибки можем получить при запуске демо-версии, и скожем ли их побороть. А ещё надо чтобы фронт-энд приспособился под новое решение.

Получится ли адаптировать за 3 месяца существующий фронтэнд и бэкэнд от премиумчесса с использованием джава7\хибернейт3– сложно сказать. Бэкэнд от премиумчесса – это тоже, фактически, сильно устаревшее решение, требующее рефакторинга и распиливания на отдельные модули. Написать новый собственный бэкэнд (спринг5\джава14\хибернейт5) за 3 месяца, который полностью продублирует весь бэкэнд премиумчесса, с разработкой новой структуры БД под постгрес – тоже малореально. Повторить только базовую часть (игроки, партии, турниры), с новым рестфул апи, новой БД под посгре12, с юнит-тестами, с конкретными требованиями по нагрузке на каждый рестфул сервис, с нагрузочным тестированием – это вполне возможно.

Мое субъективное мнение – 3 месяца - это мало для обоих вариантов. Если адаптировать и развивать решение c джава7\хибернейт3\спринг2.5, то, фактически, это развитие устаревшего решения на старых технологиях. И не факт, что успеем. Все зависит от того, в какие проблемы упремся при доработках под существующий фронт-энд, и сможем ли быстро их преодолеть.

На старых технологиях много ручного труда и рутинного кодирования, как следствие, сложней отладка и больше ошибок. Спринг5\хибернейт5\джава8 многое берут на себя и прячут под капот, там меньше кодирования, меньше ошибок, меньше сроки разработки.

Сейчас надо определиться, каким путем пойти, через адаптацию\развитие старого или написание нового. Думаю, что в обоих вариантах есть риски. Разбираться со старым и адаптировать его под существующий фронт, или писать новое с оглядкой на старые сорсы — тут сложно сказать, что окажется быстрей и эффективней.

Самый плохой вариант, когда не успеем адаптировать старое под жавой7\спринг2.5 за 3 месяца и будем год ещё дописывать\устранять ошибки под существующий фронт. Если писать новое под джаву14\спринг5, то на руках через 3 месяца у нас гарантировано будет будет рабочий пилот, который реализует лишь какой-то базовый функционал. Считаю, что за год его реально довести до продакшена, а также последовательно развивать его в будущем согласно возникающих требований.

============================================

1. Нам нужны исходники Core-Server?

+++ Мы не видим что там внутри, а не зная внутрянки сложно понять, понадобится их менять в будущем или нет.  
2. Почему мы можем не успеть впилить протокол в текущие исходники?

+++ Не совсем понял вопроса. Протокол, как я понял, — это существующий набор рест-запросов (post, get). Старый фронт заточен под один пул запросов, пул запросов от премиумчесса ничего не знает про существующий фронт-энд и его потребности. Их надо друг к другу адаптировать.

3. Вариант с прокси сервером может оказаться предпочтительным?

+++ Неизвестно, надо пробовать. Если говорить субъективно, то лично мне не нравится такой микс, вряд ли этот подпроект удастся задействовать в будущем.

//==========================================

«desks — leg» - это имеется ввиду «legacy» ? Устаревшее - «нужно» (требуется рефакторинг) или устаревшее «не нужно»?

«matches old» - «old» = не нужно?

А что со статусом следующих секций (не упомянуты вообще):

- auth

- banner

- block

- cameras

- charge

- coupon

- gaming-banner

- me

- mediafile

- news-images

- news

- payment

- plans

- selling

- subscriptions

- team-players

- widgets

//=================================================

Судя по приведенному списку группы запросов от [Pavel Chikichev](https://app.slack.com/team/U018RPC6KB2) , в старом проекте активно использовались следующие сущности:

* players
* rating
* timecontrols
* tours
* online

Т.е. всего 5 штук?

Т.е. по сути актуально всего 5 разделов из общего списка?

* players
* rating
* timecontrols
* tours
* me
* online

//=================================================

Была проведена пробная модернизация проекта - тестовое использование джава8 для компиляция проекта game-server (premiumChess).

Было добавлено в проект использование следующих библиотек Lombok 1.8.12 , JPA 2.2, SpringFramework 5.2.7 (Core, Reactor, Security, Data, Boot, WebFlux, Test)

Пробная компиляция проекта выявила проблемы совместимости используемой библиотеки Spring Test 2.5 и Spring Test 5.2.7. Некоторые классы для тестирования, используемые в Spring Test 2.5 были физически удалены из Spring Test 5.2.7. Для исправления ошибок совместимости следует доработать или переписать тесты под классы Spring Test 5.2.7 или временно отключить эти тесты.

//================================================

- исключено использование библиотек спринг 2.5, (core, test, полностью перешли на спринг 5.2.7)

- доработаны старые тесты для использования спринг 5.2.7 (вместо 2.5), подключены новые классы из версии 5.2.7, работа модернизированных тестов пока не проверялась, т. к. пока нет всего нужного окружения

- перешли с хибернейт 3.3.2 на 5.4.17

- устранены ошибки компиляции, которые появились после перехода на хибернейт 5.4.17. (приведение списковых типов, финализация переменных, использующих коллекции)

- разработан тест использующий Spring WebFlux + Spring Boot ( реактивный RESTful сервис и клиент)

Текущее состояние: проект полностью компилируется и собирается под джава8+спринг5+хибернет5.4.

Текущая проблема: не инициализируется окружение спринг-бут приложения в режиме запуска теста (не работает @Autowired), разбираюсь с этим; проверить работу новых классов из новой платформы спринг-5 (SecurityWebFilterChain, RouterFunction) пока не удалось из-за ошибки инициализации спринг-бут приложения

//===============================================

- полностью отказались от Spring 2.5 и перешли на Spring 5.2.7

- доработана возможность запуска Spring Boot приложения

- добавлено использование библиотек spring-data-jpa, spring-reactor, reactor-core, undertow, spring-tx, reactor-test, spring-boot-devtools

- разработан демо-тест RESTful API с использованием Spring WebFlux (пост запрос по адресу /api/helloSpring5)

- обновлены версии использованных библиотек mina-core, org-apache-httpcomponents, org-junit-jupiter, maven-surefire-plugin, mongo-java-driver, mina-integration; доработан исходный код под совместимость с обновленными библиотеками

- создан репозиторий обновленного проекта https://dzmitrykazyra@bitbucket.org/dzmitrykazyra/server.git