# 1. Setup

Download Spring Tool Suite ở trang này:

<https://spring.io/tools/sts/all>

Cài java core 7 và 8 (jdk 1.7 và jdk 1.8)

Workspace cho STS:



Mình mở STS ở directory này:  
C:\Users\tnguyen443\Desktop\Software tu down\spring-tool-suite-3.8.2.RELEASE-e4.6.1-win32-x86\_64\sts-bundle\sts-3.8.2.RELEASE

Trong file doc anh Hoang Dang co gui 1 chương trình cài đặt sau:



## - 2 tool trong STS

A Hoàng kiu setup 2 cái tool này để làm việc (hình như là 1 dạng SVN cho eclipse):



## - Tìm hiểu FileZilla

**FileZilla Client** (còn được gọi tắt **là FileZilla**) **là** [trình khách FTP] tự do, mã nguồn mở, đa nền tảng. Sẵn dùng cho Windows, Linux, và Mac OS X. Hỗ trợ FTP, SFTP, và FTPS. Tính đến 5 tháng 3 2009, **FileZilla**đứng thứ 5 trong số những phần mềm được tải về nhiều nhất từ SourceForge.net.

FTP là File Transfer Protocol: giao thức chuyển nhượng tập tin. Port mặc định là 21. Tuy Control Panel có giao thức transfer khác, nhưng người ta cũng dùng giao thức này, thường dùng để transfer data giữa user và host.

## Cài đặt BeyondCompare 4:



## Cài đặt lại jdk7 (jdk rộng hơn jre, vì trong máy có sẵn jre nên mình muốn setup lại jdk):



# 2. Các địa chỉ học Spring:

<https://www.tutorialspoint.com/spring/spring_hello_world_example.htm>

# 3. HelloSpring:

## 3.1 Giới thiệu:

Đây là project mình thực hiện mô phỏng đầy đủ các chức năng của Spring Framework, từ đây về sau, có gì thắc mắc thì lên Project HelloSpring kiếm trước.

<https://codersontrang.com/2013/01/20/gioi-thieu-ve-spring-framework/>

Spring nổi bật ở tính năng liên kết các thành phần lại với nhau theo cách riêng và dễ quản lý. Spring tạo ra 1 framework dễ xây dựng ứng dụng hơn J2EE.

## 3.2 Thư viện

Các thư viện cài đặt cho ví dụ Spring:



Lib dành cho cache:



Lib dành cho log:



## 3.3 Tìm hiểu về file Beans.xml - beanwiring

Beans.xml có tác dụng liên kết các thành phần trong Spring.

<https://codersontrang.com/2013/01/25/bean-wiring-cac-cach-de-lien-ket-cac-thanh-phan-trong-spring/>

Như đã biết, Spring dùng thẻ **<bean>** để khởi tạo một thành phần. Trong một thành phần này, có thể có nhiều thuộc tính tham chiếu đến khác thành phần khác được quản lý bởi Spring container. Bình thường, nếu như không dùng Spring, một thuộc tính sẽ được khai báo để tham chiếu đến một giá trị/đối tượng nào đó qua contructor hoặc phương thức setXXX() tương ứng với thuộc tính đó. Tương tự, Spring cũng cung cấp đầy đủ các cách khai báo để liên kết các thành phần qua Contructor hoặc các phương thức setter.

…

## 3.4 Properties trong java

Với những thông tin như jdbc driver, URL, port, schema…, thay vì viết trực tiếp như vậy, hoặc đặt thành những hằng số trong code thì ta có thể lưu chúng vào file cấu hình (chẳng hạn databaseConfig.properties).

Properties cũng có key và value tương tự như map, nhưng quan trọng nhất của properties là có thể năng save thành file text và load file text lên.

## 3.5 Tìm hiểu về file Beans.xml – autowiring

<https://codersontrang.com/2013/02/02/bean-autowiring-co-che-tu-dong-lien-ket-cac-thanh-phan-trong-spring/>

Qua bài viết “[Bean Wiring – các cách để liên kết các thành phần trong Spring](http://wp.me/p2A80w-5E)“, chúng ta đã được giới thiệu về các cách cơ bản mà các thành phần được khởi tạo và liên kết với nhau trong Spring thông qua việc khai báo ở file cấu hình (.xml) như thế nào. Như đã biết, chúng ta phải trực tiếp dùng thẻ <constructor-arg> hoặc <property> để chỉ ra thành phần nào sẽ được tham chiếu đến. Spring còn có một cơ chế để giảm bớt đi việc khai báo này mà vẫn đảm bảo các thành phần có thể liên kết với nhau, đó là cơ chế Bean Autowiring. Việc liên kết giữa các thành phần sẽ diễn ra hoàn toàn tự động. Cơ chế tự động được thực hiện bởi Spring container dựa vào một trong 4 cách thức sau:

* **by Name**: Tự động liên kết một thuộc tính đến một thành phần có tên trùng với tên của thuộc tính đó
* **by Type**: Tự động liên kết một thuộc tính đến một thành phần có kiểu khớp với kiểu của thuộc tính đó
* **by Constructor**: Giống như byType, nhưng thay vì tìm đến một thành phần có kiểu khớp với kiểu của tham số trong các phương thức setter thì ở đây sẽ tìm đến một thành phần có kiểu khớp với kiểu của tham số trong constructor.
* **by AutoDetect**: Tự động thực hiện cơ chế tự động liên kết *by Constructor* trước, nếu không có sự liên kết thỏa mãn, sẽ thực hiện tiếp cơ chế tự động liên kết *by Type*

## 3.6 Cơ chế Cache trong Spring Framework:

### Spring 3

<https://codersontrang.com/2013/03/04/co-che-cache-trong-spring-framework/>

Lưu ý cách này không dùng được spring 4, chỉ có thể áp dụng cho spring 3.

Các thư viện sử dụng:



File ehcache.xml – file này éo biết làm mẹ gì, nhưng có nó sẽ gây ra lỗi:

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<ehcache xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:noNamespaceSchemaLocation=*"http://ehcache.sf.net/ehcache.xsd"*>

<diskStore path=*"java.io.tmpdir"* />

<cache name=*"products"*

maxEntriesLocalHeap=*"100"*

maxEntriesLocalDisk=*"1000"*

eternal=*"false"*

timeToIdleSeconds=*"300"*

timeToLiveSeconds=*"600"*

memoryStoreEvictionPolicy=*"LFU"*

transactionalMode=*"off"*>

<persistence strategy=*"localTempSwap"* />

</cache>

</ehcache>

Lỗi mà cái file ngu ehcache.xml gây ra:



### Spring 4: chuyển qua sử dụng annotation

<http://websystique.com/spring/spring-4-cache-tutorial-with-ehcache/>

Spring4CachingExample2.zip: file này nằm trong doc của project HelloSpring, đây là ví dụ của trang web bên trên, có sử dụng down để load thư viện.

There are many caching solutions available in market today namely EhCahe, Guava Cache, Caffeine Cache, OScache, JBoss Cache,etc.

Caching Annotations:

* @Cacheable : triggers cache population
* @CacheEvict : triggers cache eviction
* @CachePut : updates the cache without interfering with the method execution
* @Caching : regroups multiple cache operations to be applied on a method
* @CacheConfig : shares some common cache-related settings at class-level
* @EnableCaching : Configuration level annotation which enables Caching

#### Các package liên quan trong ví dụ này:

com.websystique.spring.model

com.websystique.spring.service

com.websystique.spring.configuration

**File ProductServiceImpl khi không sử dụng cache:**

**package** com.websystique.spring.service;

**import** org.springframework.stereotype.Service;

**import** com.websystique.spring.model.Product;

@Service("productService")

**public** **class** ProductServiceImpl **implements** ProductService{

@Override

**public** Product getByName(String name) {

showLookupOperation();

**return** **new** Product(name, 100);

}

**public** **void** showLookupOperation(){

**try** {

**long** time = 5000L;

Thread.*sleep*(time);

} **catch** (InterruptedException e) {

**throw** **new** IllegalStateException();

}

}

}

**AppConfig khi chưa xài cache của Spring:**

**package** com.websystique.spring.configuration;

**import** java.util.Arrays;

**import** org.springframework.cache.CacheManager;

**import** org.springframework.cache.concurrent.ConcurrentMapCache;

**import** org.springframework.cache.support.SimpleCacheManager;

**import** org.springframework.context.annotation.Bean;

**import** org.springframework.context.annotation.ComponentScan;

**import** org.springframework.context.annotation.Configuration;

@Configuration

@ComponentScan(basePackages = "com.websystique.spring")

**public** **class** AppConfig {

@Bean

**public** CacheManager cacheManager() {

// configure and return an implementation of Spring's CacheManager SPI

SimpleCacheManager cacheManager = **new** SimpleCacheManager();

cacheManager.setCaches(Arrays.*asList*(**new** ConcurrentMapCache("products")));

**return** cacheManager;

}

}

**SampleApplication khi chưa xài cache của Spring:**

**package** com.websystique.spring.configuration;

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

**import** org.springframework.context.annotation.AnnotationConfigApplicationContext;

**import** org.springframework.context.support.AbstractApplicationContext;

**import** com.websystique.spring.service.ProductService;

**public** **class** SampleApplication {

**private** **static** **final** Logger ***logger*** = LoggerFactory.*getLogger*(SampleApplication.**class**);

**public** **static** **void** main(String[] args) {

AbstractApplicationContext context = **new** AnnotationConfigApplicationContext(AppConfig.**class**);

ProductService service = (ProductService) context.getBean("productService");

***logger***.info("Iphone ->"+service.getByName("Iphone"));

***logger***.info("Iphone ->"+service.getByName("Iphone"));

***logger***.info("Iphone ->"+service.getByName("Iphone"));

((AbstractApplicationContext) context).close();

}

}

Nếu không có cache, phải 5 giây thì chương trình mới lấy được data:



Lib dành cho log:



Lib dành cho cache:



Không hiểu sao cache dùng cho ví dụ này vẫn k xài được.

Trong lúc sử dụng trên trang chủ Spring có nói đến Gradle, cải tiến từ Ant và Maven, khuyên mình đọc sách này: Gradle in Action

### Spring 4: Ví dụ trên trang chủ Spring:

<https://spring.io/guides/gs/caching>

Giới thiệu khái niệm cache trong spring:

<http://dangtritue.blogspot.com/2008/09/xy-dng-h-thng-cache-vi-spring-aop.html>

Thư viện mình dùng maven để down được, chạy được luôn:



# 4. Maven Project

Bà nội nó vẫn k làm được cái cache trong spring.

<http://o7planning.org/vi/10127/huong-dan-lap-trinh-spring-cho-nguoi-moi-bat-dau#a720908>

Dùng cái hướng dẫn này mình sử dụng được file pom để down các thư viện về

# 5. Các lưu ý cho Project Daito:

\_ Column nếu là String thì k được NULL, nếu NULL thì thêm ký tự single space vào.

\_ Column nếu là numberic thì k được NULL, nếu NULL thì thêm số 0 vào.

## EBCDIC:

(Extended Binary Coded Decimal Interchange Code). Nó là mã có 8 cột, không có cột kiểm tra (như bảng 1.4).

|  |  |  |
| --- | --- | --- |
| **Table of EBCDIC character set Document Produced by Jim Plaxco** | | |
| decimal value | hex value | Character |
| 000 | 00 | NUL |
| 001 | 01 | SOH |
| 002 | 02 | STX |
| 003 | 03 | ETX |
| 004 | 04 | PF |
| 005 | 05 | HT |
| 006 | 06 | LC |
| 007 | 07 | DEL |
| 008 | 08 | GE |
| 009 | 09 | RLF |
| 010 | 0A | SMM |
| 011 | 0B | VT |
| 012 | 0C | FF |
| 013 | 0D | CR |
| 014 | 0E | SO |
| 015 | 0F | SI |
| 016 | 10 | DLE |
| 017 | 11 | DC1 |
| 018 | 12 | DC2 |
| 019 | 13 | TM |
| 020 | 14 | RES |
| 021 | 15 | NL |
| 022 | 16 | BS |
| 023 | 17 | IL |
| 024 | 18 | CAN |
| 025 | 19 | EM |
| 026 | 1A | CC |
| 027 | 1B | CU1 |
| 028 | 1C | IFS |
| 029 | 1D | IGS |
| 030 | 1E | IRS |
| 031 | 1F | IUS |
| 032 | 20 | DS |
| 033 | 21 | SOS |
| 034 | 22 | FS |
| 035 | 23 |  |
| 036 | 24 | BYP |
| 037 | 25 | LF |
| 038 | 26 | ETB |
| 039 | 27 | ESC |
| 040 | 28 |  |
| 041 | 29 |  |
| 042 | 2A | SM |
| 043 | 2B | CU2 |
| 044 | 2C |  |
| 045 | 2D | ENQ |
| 046 | 2E | ACK |
| 047 | 2F | BEL |
| 048 | 30 |  |
| 049 | 31 |  |
| 050 | 32 | SYN |
| 051 | 33 |  |
| 052 | 34 | PN |
| 053 | 35 | RS |
| 054 | 36 | UC |
| 055 | 37 | EOT |
| 056 | 38 |  |
| 057 | 39 |  |
| 058 | 3A |  |
| 059 | 3B | CUB |
| 060 | 3C | DC4 |
| 061 | 3D | NAK |
| 062 | 3E |  |
| 063 | 3F | SUB |
| 064 | 40 | BLANK |
| 065 | 41 |  |
| 066 | 42 |  |
| 067 | 43 |  |
| 068 | 44 |  |
| 069 | 45 |  |
| 070 | 46 |  |
| 071 | 47 |  |
| 072 | 48 |  |
| 073 | 49 |  |
| 074 | 4A | ¢ |
| 075 | 4B | . |
| 076 | 4C | < |
| 077 | 4D | ( |
| 078 | 4E | + |
| 079 | 4F | | |
| 080 | 50 |  |
| 081 | 51 |  |
| 082 | 52 |  |
| 083 | 53 |  |
| 084 | 54 |  |
| 085 | 55 |  |
| 086 | 56 |  |
| 087 | 57 |  |
| 088 | 58 |  |
| 089 | 59 |  |
| 090 | 5A | ! |
| 091 | 5B | $ |
| 092 | 5C | \* |
| 093 | 5D | ) |
| 094 | 5E | ; |
| 095 | 5F | ¬ |
| 096 | 60 | - |
| 097 | 61 | / |
| 098 | 62 |  |
| 099 | 63 |  |
| 100 | 64 |  |
| 101 | 65 |  |
| 102 | 66 |  |
| 103 | 67 |  |
| 104 | 68 |  |
| 105 | 69 |  |
| 106 | 6A | ¦ |
| 107 | 6B | , |
| 108 | 6C | % |
| 109 | 6D | \_ |
| 110 | 6E | > |
| 111 | 6F | ? |
| 112 | 70 |  |
| 113 | 71 |  |
| 114 | 72 |  |
| 115 | 73 |  |
| 116 | 74 |  |
| 117 | 75 |  |
| 118 | 76 |  |
| 119 | 77 |  |
| 120 | 78 |  |
| 121 | 79 | ` |
| 122 | 7A | : |
| 123 | 7B | # |
| 124 | 7C | @ |
| 125 | 7D | ' |
| 126 | 7E | = |
| 127 | 7F | " |
| 128 | 80 |  |
| 129 | 81 | a |
| 130 | 82 | b |
| 131 | 83 | c |
| 132 | 84 | d |
| 133 | 85 | e |
| 134 | 86 | f |
| 135 | 87 | g |
| 136 | 88 | h |
| 137 | 89 | i |
| 138 | 8A |  |
| 139 | 8B |  |
| 140 | 8C |  |
| 141 | 8D |  |
| 142 | 8E |  |
| 143 | 8F |  |
| 144 | 90 |  |
| 145 | 91 | j |
| 146 | 92 | k |
| 147 | 93 | l |
| 148 | 94 | m |
| 149 | 95 | n |
| 150 | 96 | o |
| 151 | 97 | p |
| 152 | 98 | q |
| 153 | 99 | r |
| 154 | 9A |  |
| 155 | 9B |  |
| 156 | 9C |  |
| 157 | 9D |  |
| 158 | 9E |  |
| 159 | 9F |  |
| 160 | A0 |  |
| 161 | A1 | ~ |
| 162 | A2 | s |
| 163 | A3 | t |
| 164 | A4 | u |
| 165 | A5 | v |
| 166 | A6 | w |
| 167 | A7 | x |
| 168 | A8 | y |
| 169 | A9 | z |
| 170 | AA |  |
| 171 | AB |  |
| 172 | AC |  |
| 173 | AD |  |
| 174 | AE |  |
| 175 | AF |  |
| 176 | B0 |  |
| 177 | B1 |  |
| 178 | B2 |  |
| 179 | B3 |  |
| 180 | B4 |  |
| 181 | B5 |  |
| 182 | B6 |  |
| 183 | B7 |  |
| 184 | B8 |  |
| 185 | B9 |  |
| 186 | BA |  |
| 187 | BB |  |
| 188 | BC |  |
| 189 | BD |  |
| 190 | BE |  |
| 191 | BF |  |
| 192 | C0 | { |
| 193 | C1 | A |
| 194 | C2 | B |
| 195 | C3 | C |
| 196 | C4 | D |
| 197 | C5 | E |
| 198 | C6 | F |
| 199 | C7 | G |
| 200 | C8 | H |
| 201 | C9 | I |
| 202 | CA |  |
| 203 | CB |  |
| 204 | CC | non-displayable |
| 205 | CD |  |
| 206 | CE | non-displayable |
| 207 | CF |  |
| 208 | D0 | } |
| 209 | D1 | J |
| 210 | D2 | K |
| 211 | D3 | L |
| 212 | D4 | M |
| 213 | D5 | N |
| 214 | D6 | O |
| 215 | D7 | P |
| 216 | D8 | Q |
| 217 | D9 | R |
| 218 | DA |  |
| 219 | DB |  |
| 220 | DC |  |
| 221 | DD |  |
| 222 | DE |  |
| 223 | DF |  |
| 224 | E0 | \ |
| 225 | E1 |  |
| 226 | E2 | S |
| 227 | E3 | T |
| 228 | E4 | U |
| 229 | E5 | V |
| 230 | E6 | W |
| 231 | E7 | X |
| 232 | E8 | Y |
| 233 | E9 | Z |
| 234 | EA |  |
| 235 | EB |  |
| 236 | EC | non-displayable |
| 237 | ED |  |
| 238 | EE |  |
| 239 | EF |  |
| 240 | F0 | 0 |
| 241 | F1 | 1 |
| 242 | F2 | 2 |
| 243 | F3 | 3 |
| 244 | F4 | 4 |
| 245 | F5 | 5 |
| 246 | F6 | 6 |
| 247 | F7 | 7 |
| 248 | F8 | 8 |
| 249 | F9 | 9 |
| 250 | FA | non-displayable |
| 251 | FB |  |
| 252 | FC |  |
| 253 | FD |  |
| 254 | FE |  |
| 255 | FF | EO |

## HSQL:

**HSQLDB là** một database đơn giản, viết bằng Java. **Là** loại database quan hệ giống MySQL, Oracle, SQL Server.

Web hướng dẫn HSQLDB:

<http://o7planning.org/vi/10287/huong-dan-cai-dat-va-cau-hinh-database-hsqldb>

<http://o7planning.org/vi/10203/cau-hinh-hsqldb-datasource-su-dung-data-source-explorer>

### Lệnh tạo table:

**CREATE** MEMORY **TABLE** PUBLIC.DEPARTMENT(DEPT\_ID **INTEGER** **NOT** **NULL** **PRIMARY** **KEY**,DEPT\_NAME **VARCHAR**(255) **NOT** **NULL**,DEPT\_NO **VARCHAR**(20) **NOT** **NULL**,LOCATION **VARCHAR**(255),**UNIQUE**(DEPT\_NO))

### Lệnh insert:

**INSERT** **INTO** DEPARTMENT **VALUES**(10,'ACCOUNTING','D10','NEW YORK')

### Table Employee với Constraint:

**CREATE** MEMORY **TABLE** PUBLIC.EMPLOYEE(EMP\_ID BIGINT **NOT** **NULL** **PRIMARY** **KEY**,EMP\_NAME **VARCHAR**(50) **NOT** **NULL**,EMP\_NO **VARCHAR**(20) **NOT** **NULL**,HIRE\_DATE **DATE** **NOT** **NULL**,IMAGE BLOB(1G),JOB **VARCHAR**(30) **NOT** **NULL**,SALARY **DOUBLE** **NOT** **NULL**,DEPT\_ID **INTEGER** **NOT** **NULL**,MNG\_ID BIGINT,**UNIQUE**(EMP\_NO),**CONSTRAINT** FK75C8D6AE269A3C9 **FOREIGN** **KEY**(DEPT\_ID) **REFERENCES** PUBLIC.DEPARTMENT(DEPT\_ID),**CONSTRAINT** FK75C8D6AE6106A42 **FOREIGN** **KEY**(EMP\_ID) **REFERENCES** PUBLIC.EMPLOYEE(EMP\_ID),**CONSTRAINT** FK75C8D6AE13C12F64 **FOREIGN** **KEY**(MNG\_ID) **REFERENCES** PUBLIC.EMPLOYEE(EMP\_ID))

Không hiểu sao không thêm data được cho table này:

### Sequence:

**CREATE** SEQUENCE AJE1FP\_SEQ START **WITH** 1;

Tạo ra 1 cái rule cho mấy loại:

SMALLINT, INTEGER, BIGINT, DECIMAL and NUMERIC

sequence generator definition

<sequence generator definition> ::= CREATE SEQUENCE [ IF NOT EXISTS ] <sequence generator name> [ <sequence generator options> ]

<sequence generator options> ::= <sequence generator option> ...

<sequence generator option> ::= <sequence generator data type option> | <common sequence generator options>

<common sequence generator options> ::= <common sequence generator option> ...

<common sequence generator option> ::= <sequence generator start with option> | <basic sequence generator option>

<basic sequence generator option> ::= <sequence generator increment by option> | <sequence generator maxvalue option> | <sequence generator minvalue option> | <sequence generator cycle option>

<sequence generator data type option> ::= AS <data type>

<sequence generator start with option> ::= START WITH <sequence generator start value>

<sequence generator start value> ::= <signed numeric literal>

<sequence generator increment by option> ::= INCREMENT BY <sequence generator increment>

<sequence generator increment> ::= <signed numeric literal>

<sequence generator maxvalue option> ::= MAXVALUE <sequence generator max value> | NO MAXVALUE

<sequence generator max value> ::= <signed numeric literal>

<sequence generator minvalue option> ::= MINVALUE <sequence generator min value> | NO MINVALUE

<sequence generator min value> ::= <signed numeric literal>

<sequence generator cycle option> ::= CYCLE | NO CYCLE

# 6 Add code cho etes

Làm nhanh quá k kịp nhìn nhưng tạm thời là vầy:

Chọn regular time

Project nhập mã code vào

Show project

Show Billing, chọn số nào đó (lúc anh Hoàng đưa thì chọn 000)