IBM Payment Manager Cassette Development Workshop

The Cookbook Exercise

Lance Bader

Choose a Cassette Name

- IBM Example - FastPay Cassette
  - Cassette name (payment type) is FastPay
  - Java jar file is FastPayCassette.jar
  - Package name is com.ibm.etill.fastpaycassette
  - Properties file is FastPay.properties
  - XML PSPL document is FastPay.en.pspl
  - Cassette class is FastPayCassette
  - Query class is FastPayCassetteQuery
  - By convention FastPay will be used a prefix to all other classes and database table names

Choose a Cassette Name

- ABC Example - FastPay Cassette
  - Cassette name (payment type) is FastPay
  - Java jar file is FastPayCassette.jar
  - Package is com.abc.ibmetill.fastpaycassette
  - Properties file is FastPay.properties
  - XML PSPL document is FastPay.en.pspl
  - Cassette class is FastPayCassette
  - Query class is FastPayCassetteQuery
  - By convention FastPay will be used a prefix to all other classes and database table names

---

Obtain and Install Materials

- JDK 1.1.8
- IBM WebSphere Payment Manager
  - IBM DB2 Universal Database
  - IBM HTTP Server
  - IBM WebSphere Application Server
- IBM WebSphere Payment Manager Service Pack
- A Professional Editor
- IBM WebSphere Payment Manager Cassette Developer Toolkit

Cookbook Assumptions

- Existing Payment Appliance
- Idempotent Appliance Transactions
- Automatic Synchronous Retry
- No Wallet
- One Account per Merchant
- One Currency per Account
- One Open Batch per Account
- Single Tier Configuration
- Automatic Open Batch with Explicit Close

Design Your Cassette

- Map Payment Manager commands to your Payment Appliance API
- Design your persistent data
  - Source of supplied parameter values
    - Account configuration
    - Protocol data
    - Data returned on previous operation
  - Evaluate returned values
    - Used on subsequent operations
    - Provides valuable feedback

Design Your Cassette *(continued)*

- Layout your Database Tables
- Design your Protocol Data
  - Choose standard framework keywords whenever possible
  - Keywords must begin with $
  - Secondary return code required for each keyword
  - Specify Java data type of value
    - Boolean, Byte Array, Integer, Long, Numeric String, Numeric Token, Path, Restricted String, String, Timestamp

---

Prepare the Base Files

- Edit and execute CreateCassette script
- Change LdbCard to your cassette name
- Change LDBCARD to your cassette name
- Change com.ibm.etill.ldbcardcassette to your package name
- Remove appliance code from Account
- Fix up scripts and shortcuts
- Build your cassette
- Add your jar file to WebSphere Application Server Classpath

Implement Your Data Model

- Update a file of SQL statements that create tables and views with your column names
- Update the PSPL tags so the Payment Manager User Interface can modify and display your data
- Make constants for your persistent data
- Update your Cassette Query class to produce the expected PSPL fields
- Update your Cassette so the validation table is correct for your protocol data keywords

---

Implement Your Account Class

- Change instance variables, setters, and getters
- Update both constructors
- Update createRecord and updateRecord
- Update modifyAccount method
- Update connect and disconnect methods
- Update initialization methods
- Payment methods will be updated later

Implement Your Batch Class

- Change instance variables, setters, and getters
- Update resurrectBatch
- Update createRecord and updateRecord
- Update closeBatch method and its associated settleBatch method in your Account class

---

Implement Your Order Class

- Change instance variables, setters, and getters
- Update constructor that uses protocol data to initialize order
- Update resurrectOrder
- Update createRecord and updateRecord
- Update acceptPayment method

Implement Your Payment Class

- Change instance variables, setters, and getters
- Update resurrectPayment
- Update createRecord and updateRecord
- Update approve method
- Update approveReversal method
- Update deposit method
- Update depositReversal method
- Update matching methods in your Account

Implement Your Credit Class

- Change instance variables, setters, and getters
- Update resurrectCredit
- Update createRecord and updateRecord
- Update refund method
- Update refundReversal method
- Update matching methods in your Account

Test Your Cassette

- Sample Store and Payment Manager user interface provide unit test
- Still need automated, reliable, repeatable test cases
  - Verify database updated correctly
  - Verify correct primary and secondary return codes are returned when error is detected
  - Verify framework states change correctly
  - Verify state checks performed correctly
  - Test on all supported platforms

Package and Ship Your Cassette

- Build and test an install package
  - Watch out for platform dependencies
  - Use information left in footprints by Payment Manager
- Ship your cassette
- Send me e-mail to tell me what you liked about the cookbook as well as what needs to be improved
- Celebrate!