oddgen – An Introduction

Philipp Salvisberg

@phsalvisberg

Agenda

1. Definitions & Concepts
2. Build & Evolve a Generator
3. Core Messages

Definitions & Concepts

What is oddgen?

our **d**ictionary-**d**riven **gen**erator

Oracle community project providing
tooling for dictionary-driven code generation

What does "dictionary-driven" mean?

The predominant part of a model is stored
in an Oracle data dictionary
or in other related relational tables.

What is a Model?

- A simplification
- With a purpose
- To be used instead of the "real thing"

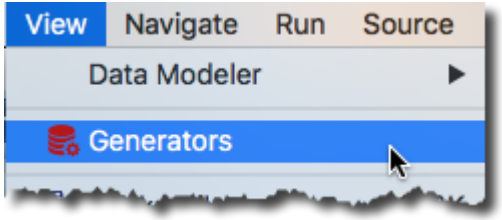
Model-Driven Software Development

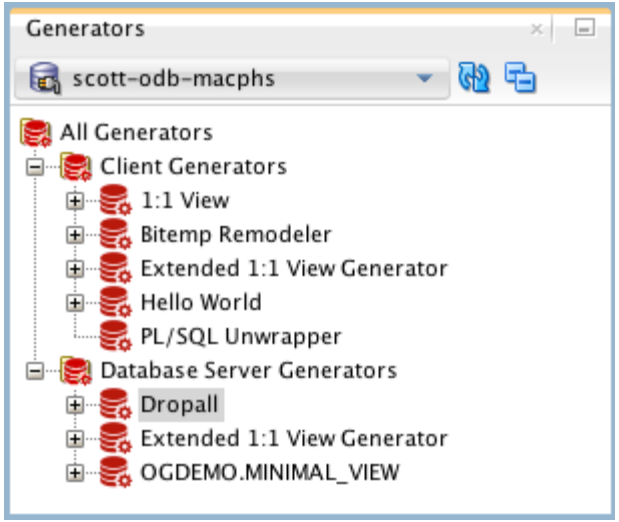
Reduce the problem-solution gap to
improve the efficency of software development.

Problem	Domain Specific Language	General Purpose Modeling Language	oddgen	High-level Programming Language	Low-level Programming Language	Machine Code
	DSL = Model	UML		Java C# Xtend Scala	Assembler C	

Oracle SQL Developer Extension

- Invokes code generators
- Adds
  - "Generators" in View menu
  - Dockable window "Generators"
  - Preference section
- Requirements
  - SQL Developer 4.0.2 or higher
  - Oracle Database Server 9.2 or higher
- Download & installation instructions
  - https://www.oddgen.org/download/





Generator Types

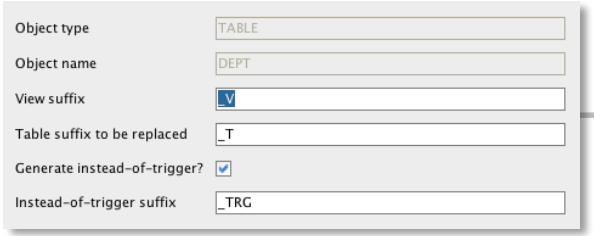
Client Generators

- Runs in the JVM of SQL Developer
- Java, Scala, Groovy, Xtend, …
- Usable for all database connections
- Autodiscovery (looking for classes in JAR files implementing the oddgen Java interface)
- Easy to use

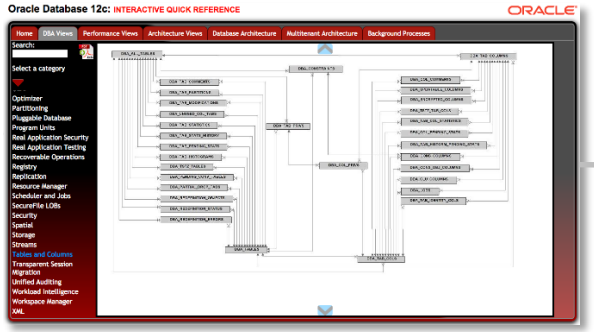
Database Server Generators

- Runs in the Database
- PL/SQL
- Usable in installed Databases only
- Autodiscovery (looking for PL/SQL packages implementing the oddgen PL/SQL interface)
- Easy to use

Generating Code



Parameters



Data Dictionary

```
val result = '''
    -- create 1:1 view for demonstration purposes
    CREATE OR REPLACE VIEW «viewName» AS
       SELECT «FOR col : columns SEPARATOR ",\n        "»«col»«ENDFOR»
         FROM «objectName»;
'''
```

Templates

Gene-rator

Code

```
1  -- create 1:1 view for demonstration purposes
2  CREATE OR REPLACE VIEW DEPT_V AS
3     SELECT DEPTNO,
4            DNAME,
5            LOC
6       FROM DEPT;
```

Model = Parameters + Data Dictonary

Template Engines

Plain PL/SQL

```
add_line('-- create 1:1 view for demonstration purposes');
add_line('CREATE OR REPLACE VIEW ' || get_view_name() || ' AS');
<<select_list>>
FOR l_rec IN c_columns
LOOP
   IF l_rec.is_first = 1 THEN
      l_line := '   SELECT ';
   ELSE
      l_line := '          ';
   END IF;
   l_line := l_line || l_rec.column_name;
   IF l_rec.is_last = 0 THEN
      l_line := l_line || ',';
   END IF;
   add_line(l_line);
END LOOP select_list;
add_line('     FROM ' || in_object_name || ';');
```

Xtend

```
val result = '''
    -- create 1:1 view for demonstration purposes
    CREATE OR REPLACE VIEW «viewName» AS
       SELECT «FOR col : columns SEPARATOR ",\n        "»«col»«ENDFOR»
         FROM «objectName»;
'''
```

FTLDB – based on Apache Freemaker

```
-- create 1:1 view for demonstration purposes
CREATE OR REPLACE VIEW ${view_name} AS
<#list columns as col>
   <#if col?is_first>
   SELECT ${col.COLUMN_NAME}<#sep>,</#sep>
   <#else>
          ${col.COLUMN_NAME}<#sep>,</#sep>
   </#if>
</#list>
     FROM ${object_name};
```

tePLSQL – based on PL/SQL Server Pages (PSP)

```
-- create 1:1 view for demonstration purposes
CREATE OR REPLACE VIEW ${view_name} AS
<% FOR l_rec IN c_columns LOOP %>
<% IF l_rec.is_first = 1 THEN %>
   SELECT <%= l_rec.column_name %><% add_comma_if(l_rec.is_last = 0); %>\\n
<% ELSE %>
          <%= l_rec.column_name %><% add_comma_if(l_rec.is_last = 0); %>\\n
<% END IF; %>
<% END LOOP; %>
     FROM ${object_name};
```

Convention over Configuration

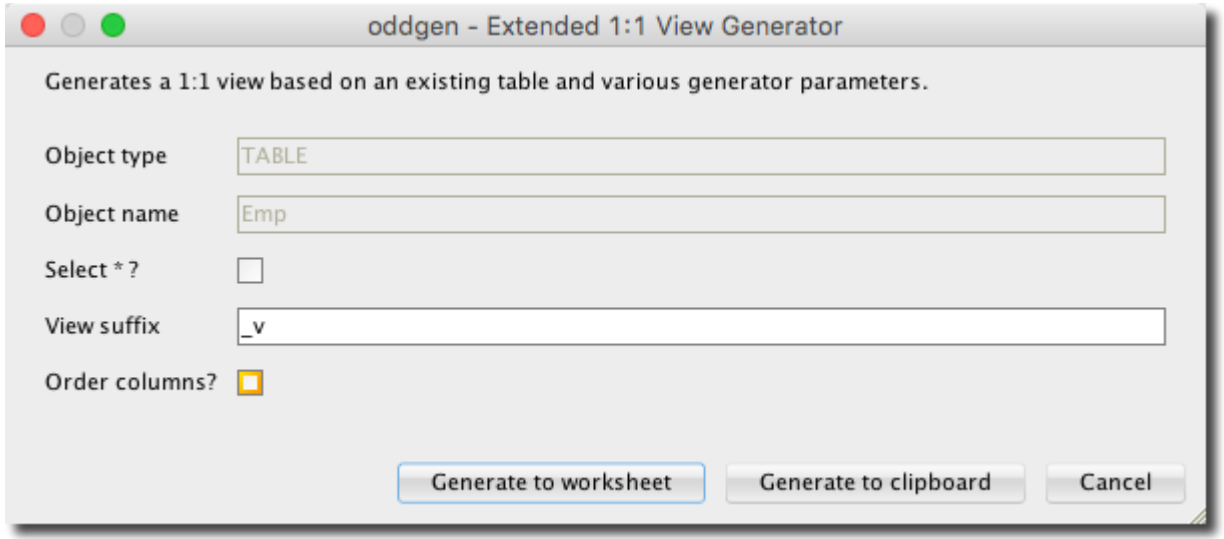
- Extensive use of sensible defaults
- No parameter entry necessary to generate useful output
- oddgen PL/SQL interface requires the implementation of just one function

```
FUNCTION generate(
   in_object_type IN VARCHAR2,
   in_object_name IN VARCHAR2
) RETURN CLOB;
```

See https://www.oddgen.org/plsql-interface/ for details.

Build & Evolve a Generator

Extended 1:1 View Generator




```
1  CREATE OR REPLACE VIEW emp_v AS
2     SELECT empno, ename, job, mgr, hiredate, sal, comm, deptno
3       FROM emp;
```

Further Information

- https://www.oddgen.org/
- https://github.com/oddgen/
- https://hub.docker.com/r/phsalvisberg/oddgendemo/

trivadis
makes IT easier.

Core Messages

Improve the Efficency of Software Development

- Reduce the problem-solution gap by using oddgen.
- Never mix generated and manually crafted code.

It Has Never Been Easier to Generate Code

- Just implement the PL/SQL package function generate to register a generator in SQL Developer.
- Implement optional functions of the oddgen PL/SQL interface to change the default behaviour.
- Have a look at the tutorials at https://www.oddgen.org/documentation/

trivadis
makes IT easier.

Questions and Answers

Philipp Salvisberg
Senior Principal Consultant

Tel. +41 58 459 52 31
philipp.salvisberg@trivadis.com

trivadis
makes IT easier.