JavaFX Introduction

Why and What is JavaFX?

Per Lundholm

Goals

- Motivation for a language such as JavaFX
- Architecture
- The JavaFX language

Why JavaFX?

- Why was it created?
- Why should you spend time with JavaFX?

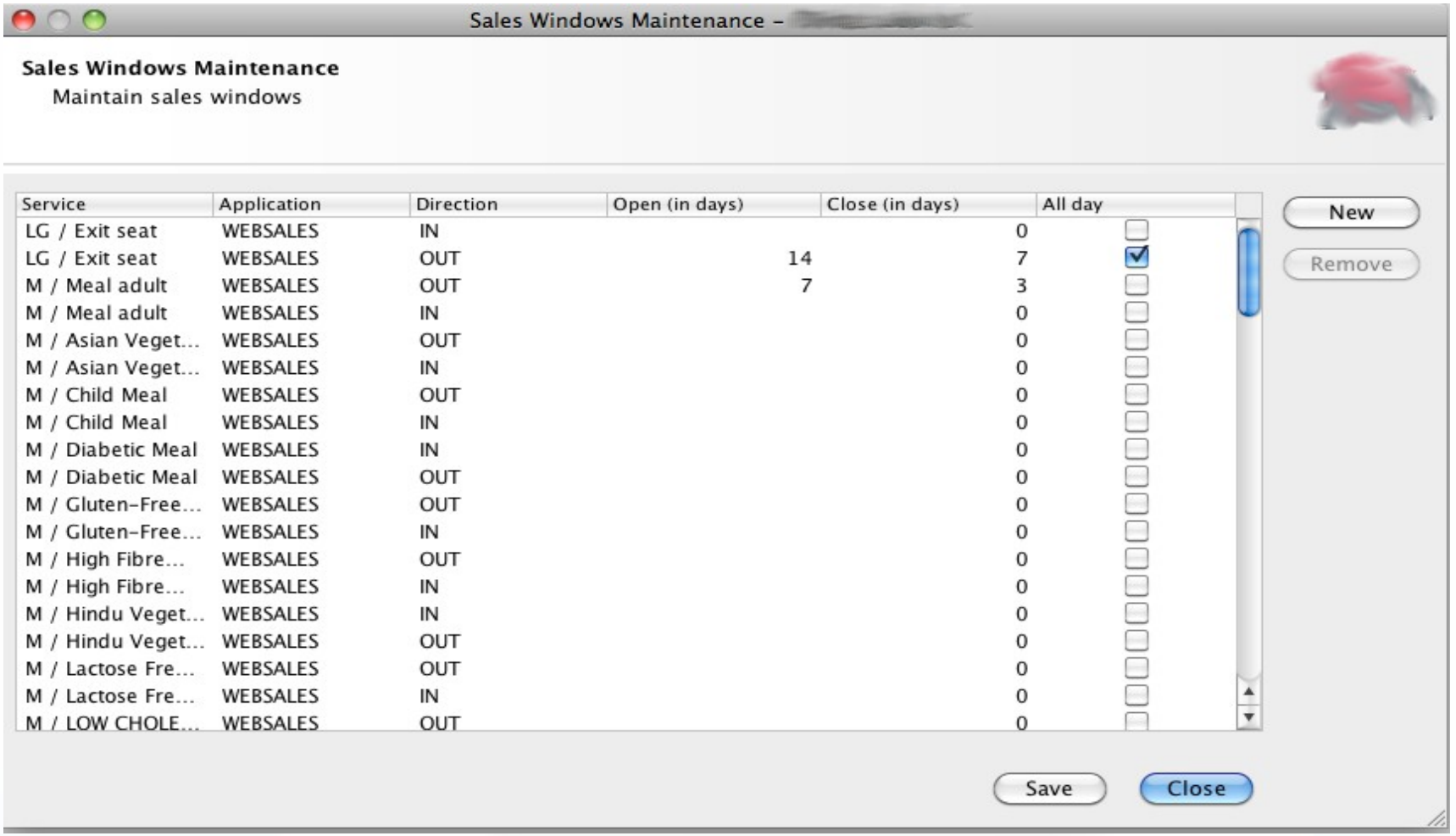
Why was JavaFX Created?

- New styles of interactive application
  - So called richer interfaces
    - Animations
    - Effects
    - Innovative new ways of doing interactions
  - Java Swing is not up to it and frankly no fun
- A language specialized for GUI that still have all that Java

Why Learn JavaFX

- Personal opinion – it is fun!
  - Discover new ways of designing user interfaces
  - Excel is for numbers, JavaFX for simulation
- JavaFX runs on everything Java
  - Focuses on the GUI, leaving the rest to Java
- More promising, technically, than Flash, Silverlight and GWT for app

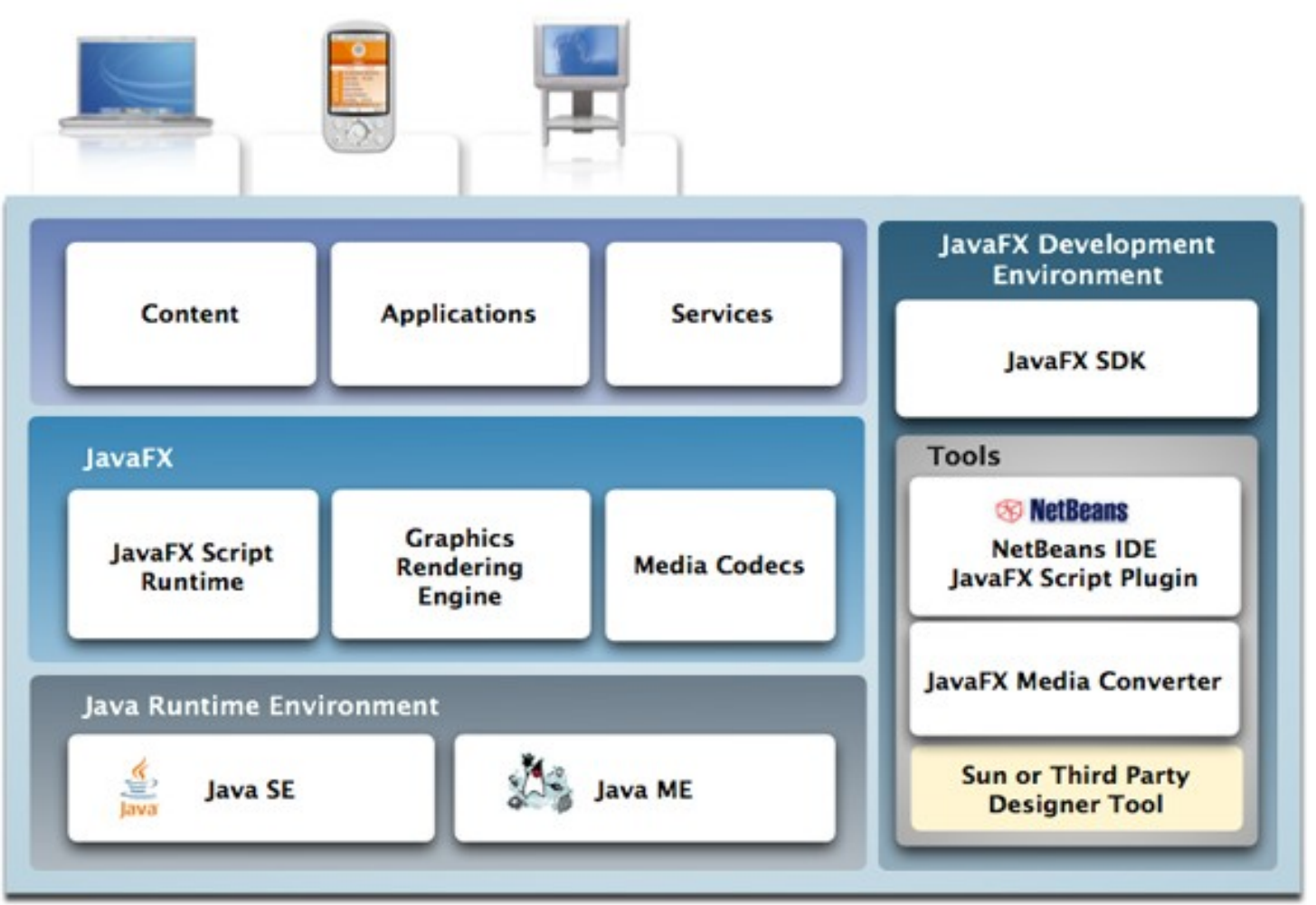
New Ways - Why?



crisp

Per Lundholm

Architecture



JavaFX Language

- What makes it a GUI language?
  - Declarative
    - GUI is described as a graph
  - Variable bind and triggers
  - Animation support
  - Built-in effects for reflections, shadows, etc
  - Built-in transformations
  - Parser for XML & JSon to reach services

crisp

Per Lundholm

Language Details: Datatypes

- The usuals suspects: strings, integers, floats et al
- Extra: Duration and Sequence
  - Duration express a period of time, from ms to hour
    - E.g. 15ms, 25s, 24h
  - Sequences are more powerful than arrays
    - var cards = [[2..10], "J", "Q", "K", "A"];
    - cards [x | cardValue(x) > 10];
    - function cardValue(card: Object) {
    - for(c in cards) { if(c == card) { return indexof c + 2; } }
    - throw IllegalArgumentException {};
    - }

Per Lundholm

Language: Objects

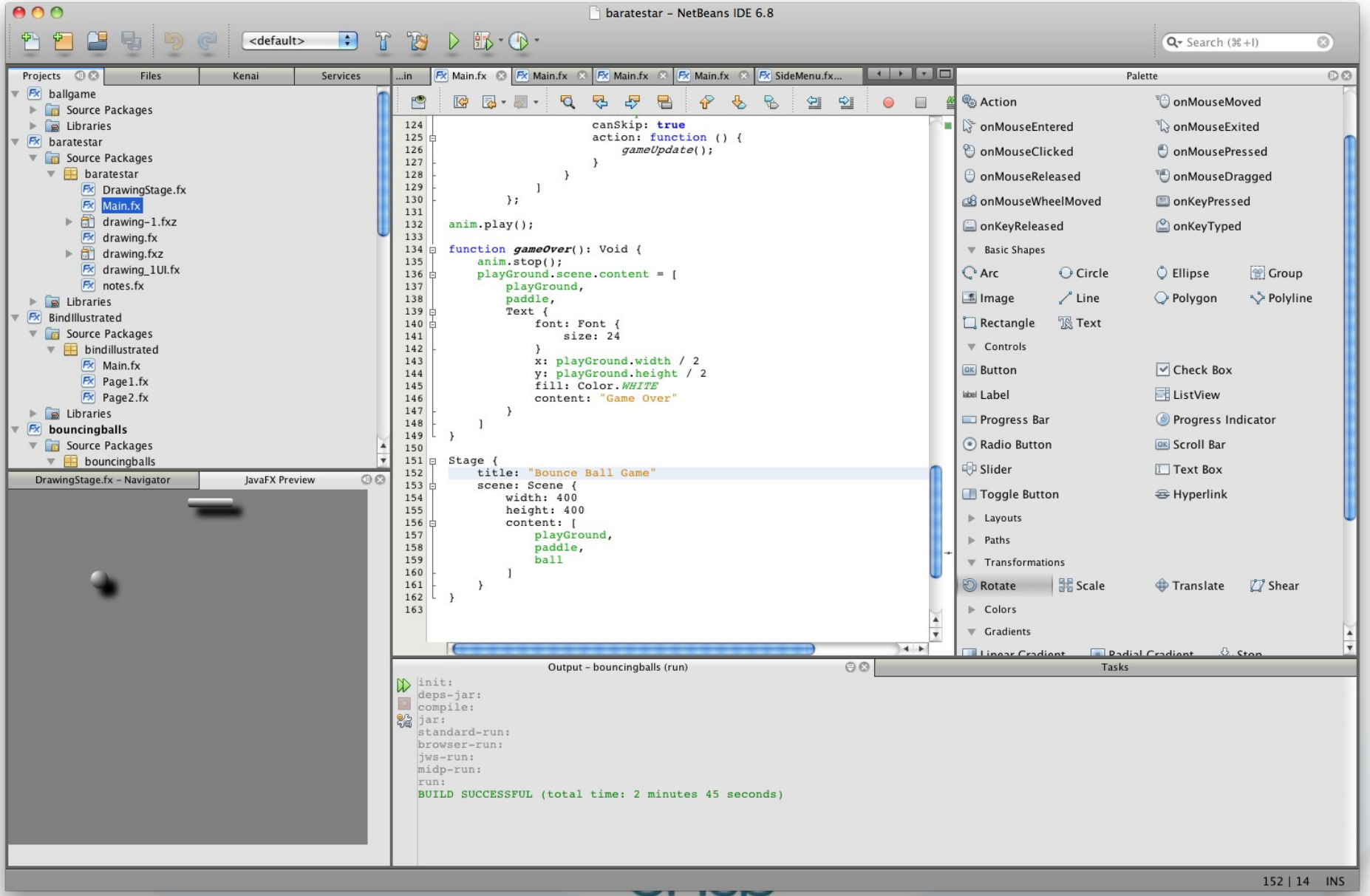
- Like Java, there are classes and objects
- No interfaces, but possible to implement a Java interface in JavaFX (typically callbacks)
- Extra: mixin inheritance
- Object initialization adapted for GUI
  - var obj = SomeClass { namedParam : value }
  - GUI classes tend to have many properties!

Language: Functions

- Functions are first-class objects
  - Thus can be a parameter to a function

```
function manipulate(s:String , f:function(:String):String) : Void {
  println("{s} = "+ f(s));
}
function lower(s:String) : String {
  s.toLowerCase();
}
function sub(s:String) : String {
  s.substring(0,4);
}
manipulate("JavaFX" , lower);  -> javafx
manipulate("JavaFX" , sub); -> Java
```

NetBeans Quickly



Per Lundholm