PROGRAMACIÓN

DECLARATIVA

Grado en Ingeniería Informática

Campus de Móstoles. URJC

Curso 2022-2023

Contenido

Datos básicos

- Asignatura obligatoria, 6 créditos, 1er cuatrimestre,
  3er curso (GII, GII+GIS, GII+GMAT)
  4to curso (GII+GADE, GII+GIC)
- Horarios:
  - Lunes de 13 a 15.
  - Viernes de 11 a 13.

Ubicación en el Plan de Estudios

1. Forma parte de la materia **Lenguajes de Programación**, junto con la asignatura
   - “Procesadores de Lenguajes” (3º, 2º cuatrimestre)
2. Tiene como requisito previo recomendado las asignaturas:
   - “Lógica” y “Matemática Discreta y Álgebra” (1º, 1er cuatrimestre)
   - “Introducción a la Programación” (1º, 1er cuatrimestre)
   - “Estructura de Datos” (1º, 2do cuatrimestre)
   - “Análisis y diseño de Algoritmos” (2º, 2do cuatrimestre)

Profesorado

Programación funcional

- Juan Manuel Serrano (juanmanuel.serrano@urjc.es, despacho 024, Departamental II, Móstoles).
- Apoyo prácticas: Joaquín Arias (joaquin.arias@urjc.es).

Programación lógica

- Ana Pradera (ana.pradera@urjc.es, despacho 110, Departamental II, Móstoles).
- Apoyo prácticas: Joaquín Arias (joaquin.arias@urjc.es).

Tutorías: concertar cita por correo electrónico.

Descripción

Programación Declarativa

- Se trata de un paradigma de programación en el que los programas se limitan a declarar o describir el problema que se pretende resolver (la componente lógica), sin especificar ninguna estrategia para su resolución (la componente de control).
- A diferencia de los programas imperativos, los programas declarativos describen exclusivamente qué hay que resolver, sin entrar en *cómo* hacerlo (es decir, sin establecer secuencias de instrucciones, gestión de memoria, ...).
- La programación declarativa consta de varios subparadigmas. Los dos principales son la programación funcional y la programación lógica.

Programación Funcional

- Se introdujo en la década de 1950 (lenguaje Lisp) y tiene sus raíces en la teoría matemática del Cálculo Lambda (sistema formal lógico, basado en funciones matemáticas, propuesto por Church y Kleene en la década de 1930 como modelo de computación para el estudio de la computabilidad).

- Funcionamiento básico:
  - **Representación del conocimiento**: mediante ecuaciones que describen funciones matemáticas (relaciones que asocian una única salida a cada conjunto de entradas).
  - **Computación**: mediante la evaluación de expresiones construidas combinando funciones descritas previamente.

- Lenguaje de programación funcional que se va a utilizar:

SCALA

Programación Lógica

- Se introdujo en la década de 1970 (lenguaje Prolog) y tiene sus raíces en los campos de la Lógica Matemática (ciencia que estudia la validez de los razonamientos) y la Demostración Automática (búsqueda de sistemas para demostrar la validez de razonamientos que se puedan ejecutar de forma eficiente en un ordenador).
- Funcionamiento básico:
  - **Representación del conocimiento**: mediante fórmulas lógicas que describen predicados lógicos (estos permiten expresar *propiedades* de objetos o *relaciones* entre ellos).
  - **Computación**: mediante la aplicación de un sistema de demostración automático que permite averiguar si una determinada fórmula es o no consecuencia lógica de un conjunto de fórmulas descritas previamente y computar soluciones.
- Lenguaje de programación lógica que se va a utilizar:

PROLOG

Características básicas de la programación declarativa

- Programación de alto nivel con base matemática.
- Uso extenso de la recursión (no hay instrucciones de repetición).
- Estructura de datos fundamental: listas de elementos.
- Gestión automática de memoria.
- Fácil implementación y manejo de construcciones de orden superior (aquellas en las que las funciones/predicados pueden ser pasados como argumentos o devueltos como salida), facilitando así la programación genérica y la reutilización de código.
- Programación sin efectos laterales (transparencia referencial).

Los programas declarativos son:

- Concisos, potentes y fiables.
- Fáciles de entender, mantener, modificar, extender, verificar y paralelizar.

Objetivos

Al terminar el curso, los alumnos deberán:

1. Conocer los fundamentos teóricos, el funcionamiento, las características básicas, las aplicaciones y la evolución del paradigma de la Programación Declarativa, en particular, de la programación funcional y de la programación lógica.
2. Comprender la forma de operar, los rudimentos y las técnicas básicas del lenguaje funcional SCALA y del lenguaje lógico PROLOG.
3. Ser capaces de utilizar los lenguajes SCALA y PROLOG para la resolución de problemas sencillos.

Temario Programación Funcional

PF-1

El paradigma de programación funcional: declaratividad. Lenguajes de programación funcional: Scala. La programación funcional en la industria del software.

PF-2

Tipos algebraicos de datos y funciones. Programación genérica. Isomorfismo Curry-Howard. Recursividad.

PF-3

Funciones de orden superior y esquemas de recursión: catamorfismos.

Temario Programación Lógica

PL-1. El paradigma de la programación lógica.

Fundamentos teóricos, evolución histórica, funcionamiento, características básicas, aplicaciones, ...

PL-2. El lenguaje Prolog: aspectos básicos

Características generales, sintaxis (predicados, programas, consultas), semántica (unificación, regla y árboles de resolución), predicados predefinidos básicos (clasificación y comparación de términos, aritmética, entrada/salida), manejo de listas, predicados de control (corte).

PL-3. El lenguaje Prolog: aspectos avanzados

Negación, recolección de todas las soluciones, uso e implementación de predicados de orden superior (aplicación, filtrado, plegado, ...).

Bibliografía básica Programación Funcional

- Martin Odersky, Lex Spoon, and Bill Venners
  Programming in Scala. A comprehensive step-by-step guide
  Third Edition. Artima.

- Paul Chiusano and Runar Bjarnason.
  Functional Programming in Scala.
  Manning.

- Alvin Alexander.
  Functional Programming, Simplified: (Scala Edition).
  `https://alvinalexander.com/scala/functional-progra`

Bibliografía básica Programación Lógica

- W.F. Clocksin and C.S. Mellish.
  Programming in Prolog
  Springer-Verlag, Berlin, fifth edition, 2003.

- L. Sterling and E. Shapiro.
  The Art of Prolog
  The MIT Press, Cambridge, Mass., second edition, third print, 1999.

- I. Bratko.
  Prolog Programming for Artificial Intelligence
  Addison-Wesley, Reading, Massachusetts, third edition, 2001.

Tipo de evaluación

- La evaluación se llevará a cabo mediante un único examen, **presencial**, dividido en dos partes:

Partes	Temas	Mínimo	Reevaluable	Peso
PF	Prog. Func.	4	Sí	50 %
PL	Prog. Lóg.	4	Sí	50 %

- La nota final se calculará como sigue:

$$\text{Nota} = \begin{cases} 0,5 * PF + 0,5 * PL & \text{si } PF \geq 4 \text{ y } PL \geq 4 \\ min(4; 0,5 * PF + 0,5 * PL), & \text{en otro caso} \end{cases}$$

- Para aprobar la asignatura será necesario tener, además de al menos un 4 sobre 10 en cada una de las partes, una nota media igual o mayor que 5 sobre 10.

Convocatorias

- Habrá dos convocatorias, que se realizarán en las fechas establecidas por la Universidad:
  - enero-febrero (convocatoria ordinaria);
  - junio-julio (convocatoria extraordinaria).
- En cada convocatoria se incluirán las dos partes PF y PL descritas más arriba. En la convocatoria ordinaria, la primera prueba, PF, se realizará durante el curso, al terminar la parte del temario correspondiente.
- Los alumnos que obtengan una calificación igual o superior a 4 puntos en alguna de las partes en la primera convocatoria no tendrán que repetir esa parte en la segunda convocatoria (en caso de que lo hagan, se entenderá que renuncian a la calificación obtenida en la primera convocatoria).

Planificación

Clases: L 13-15, V 11-13

- Primer día de clase: viernes 16 de septiembre de 2022.
- Último día de clase: lunes 19 de diciembre de 2022.

Realización de pruebas (presenciales)

- Convocatoria ordinaria:
  - Parte PF (programación funcional): L 31-10-2022, de 13 a 15
  - Parte PL (programación lógica): en la fecha oficial establecida por la URJC (fecha provisional: 23-1-2023, 12:00).
- Convocatoria extraordinaria: ambas partes se realizarán en la fecha oficial establecida por la URJC (fecha provisional: 29-6-2023, 9:00).

¿¿ PREGUNTAS ??

PROGRAMACIÓN DECLARATIVA

PROGRAMACIÓN LÓGICA

PRESENTACIÓN DE LA MATERIA

Con acceso a todo el material docente:
presentaciones, planificación, bibliografía, ejercicios, prácticas
y exámenes (con sus correspondientes soluciones)

Grado en Ingeniería Informática URJC

Curso 2022-2023

Ana Pradera

Contenido

INTRODUCCIÓN

- Esta presentación permite acceder a todo el material docente propuesto para el estudio de la parte de **Programación Lógica** de la asignatura *Programación Declarativa*, impartida en el tercer curso del Grado en Ingeniería Informática de la Universidad Rey Juan Carlos.

- Después de una breve presentación de la materia, se incluye un índice de contenidos con enlaces a las presentaciones correspondientes, una propuesta de planificación temporal para su estudio, unas breves instrucciones para la realización de las prácticas, una colección de exámenes resueltos y, por último, una bibliografía.

- La materia de Programación Lógica se imparte *después* de Programación Funcional, por lo que se parte del hecho de que el alumnado ya está familiarizado con conceptos básicos de la Programación Declarativa como el manejo de listas, la recursión, la recursión de cola o las construcciones de orden superior.

- Las presentaciones incluyen, en los lugares oportunos para su realización, tanto ejercicios como enlaces a las prácticas.

- En ambos casos se proponen soluciones, aunque es muy recomendable usar estas soluciones propuestas solo para cotejarlas con las soluciones propias, *una vez hechos los ejercicios*.

PRESENTACIÓN Y OBJETIVOS

La Programación Lógica es, junto con la **Programación Funcional**, uno de los principales subparadigmas de la Programación Declarativa.

Programación Declarativa

- Se trata de un paradigma de programación en el que los programas se limitan a declarar o describir los problemas (la componente lógica, el "qué"), omitiendo cualquier estrategia para su resolución (la componente de control, el "cómo").
- Es un paradigma de alto nivel, con base matemática, que hace un uso extenso de la recursión y de construcciones de orden superior (aquellas en las que las funciones/predicados pueden ser pasados como argumentos o devueltos como salida), facilitando así la programación genérica y la reutilización de código.
- Los programas declarativos son concisos, potentes y fiables.

Programación Lógica

- Se introdujo en la década de 1970 y tiene sus raíces en los campos de la Lógica Matemática (ciencia que estudia la validez de los razonamientos) y la Demostración Automática (búsqueda de sistemas para demostrar la validez de razonamientos que se puedan ejecutar de forma eficiente en un ordenador).
- Funcionamiento básico:
  - **Representación del conocimiento**: mediante fórmulas lógicas que describen predicados lógicos (estos permiten expresar *propiedades* de objetos o *relaciones* entre ellos).
  - **Computación**: mediante la aplicación de un sistema de demostración automático que permite averiguar si una determinada fórmula es o no consecuencia lógica de un conjunto de fórmulas descritas previamente y computar las soluciones.
- Lenguaje y entorno de programación que se van a utilizar:

  PROLOG y SWISH, herramienta online de SWI-Prolog

Objetivos

Al terminar el curso, las/os alumnas/os deberán:

1. Conocer la evolución, los fundamentos teóricos, el funcionamiento, las características básicas y las principales aplicaciones del paradigma de la Programación Lógica.
2. Conocer y comprender los rudimentos y las técnicas básicas del lenguaje lógico PROLOG. En particular:
   - Su sintaxis y su semántica operacional (método de cómputo).
   - Sus herramientas para clasificación y comparación de términos, aritmética, entrada/salida, manejo de listas y control (predicado de corte).
   - Algunos aspectos más avanzados: el predicado de negación, predicados para recolección de soluciones y uso e implementación de predicados de orden superior.
3. Ser capaces de utilizar todo lo anterior para la resolución de problemas sencillos.

ÍNDICE (con enlaces a todo el material)

TEMA PL3. EL LENGUAJE PROLOG, ASPECTOS MÁS AVANZADOS

PLANIFICACIÓN

- La materia de Programación Lógica tiene asignados 3 créditos ECTS, que se corresponden con unas 3*25=75 horas de trabajo de las/os estudiantes, de las cuales:
  - Entre 26 y 28 horas son de asistencia a clases, teóricas y prácticas, impartidas en 13 o 14 sesiones de 2 horas cada una.
  - El resto (49 o 47 horas) son para trabajo fuera del aula, dedicado a repasar las presentaciones, profundizar en los distintos temas consultando la bibliografía y resolver ejercicios, tanto los propuestos en las presentaciones y en las prácticas como ejercicios adicionales procedentes de libros o Internet.
- En esta tabla se propone una planificación temporal para el estudio de la materia dividida en 14 sesiones de 2 horas, facilitando para cada una de ellas los contenidos y objetivos cubiertos, los materiales y prácticas para su estudio y el tiempo mínimo de trabajo fuera del aula recomendado.

INSTRUCCIONES PARA LA REALIZACIÓN DE LAS PRÁCTICAS

- Las prácticas se realizan mediante los notebooks ofrecidos por la herramienta online de SWI-Prolog, SWISH.
- Aquí puede encontrar unas instrucciones al respecto.

EXÁMENES RESUELTOS

En este documento tiene a su disposición algunos exámenes resueltos.

BIBLIOGRAFÍA

Todas las presentaciones, ejemplos, ejercicios y prácticas que componen este material docente están basados en fuentes diversas, en particular en los libros y recursos que se citan a continuación.

Bibliografía básica

- L. Sterling and E. Shapiro.
  The Art of Prolog.
  The MIT Press, Cambridge, Mass., second edition, 1994.
- W.F. Clocksin and C.S. Mellish.
  Programming in Prolog.
  Springer-Verlag, Berlin, fifth edition, 2003.
- I. Bratko.
  Prolog Programming for Artificial Intelligence.
  Addison-Wesley, Reading, Massachusetts, third edition, 2001.

Bibliografía complementaria

- Foundations of Logic Programming (Second Edition), John Lloyd, Springer-Verlag, 1987.
- The Craft of Prolog, R. O'Keefe, The MIT Press, Cambridge, MA, 1990.
- Logic, Programming and Prolog, Ulf Nilsson and Jan Maluszynski, John Wiley & Sons Ltd, 1996.
- SWI-Prolog, entorno de programación en Prolog de dominio público.
- comp.lang.prolog. Faq
- Association for Logic Programming

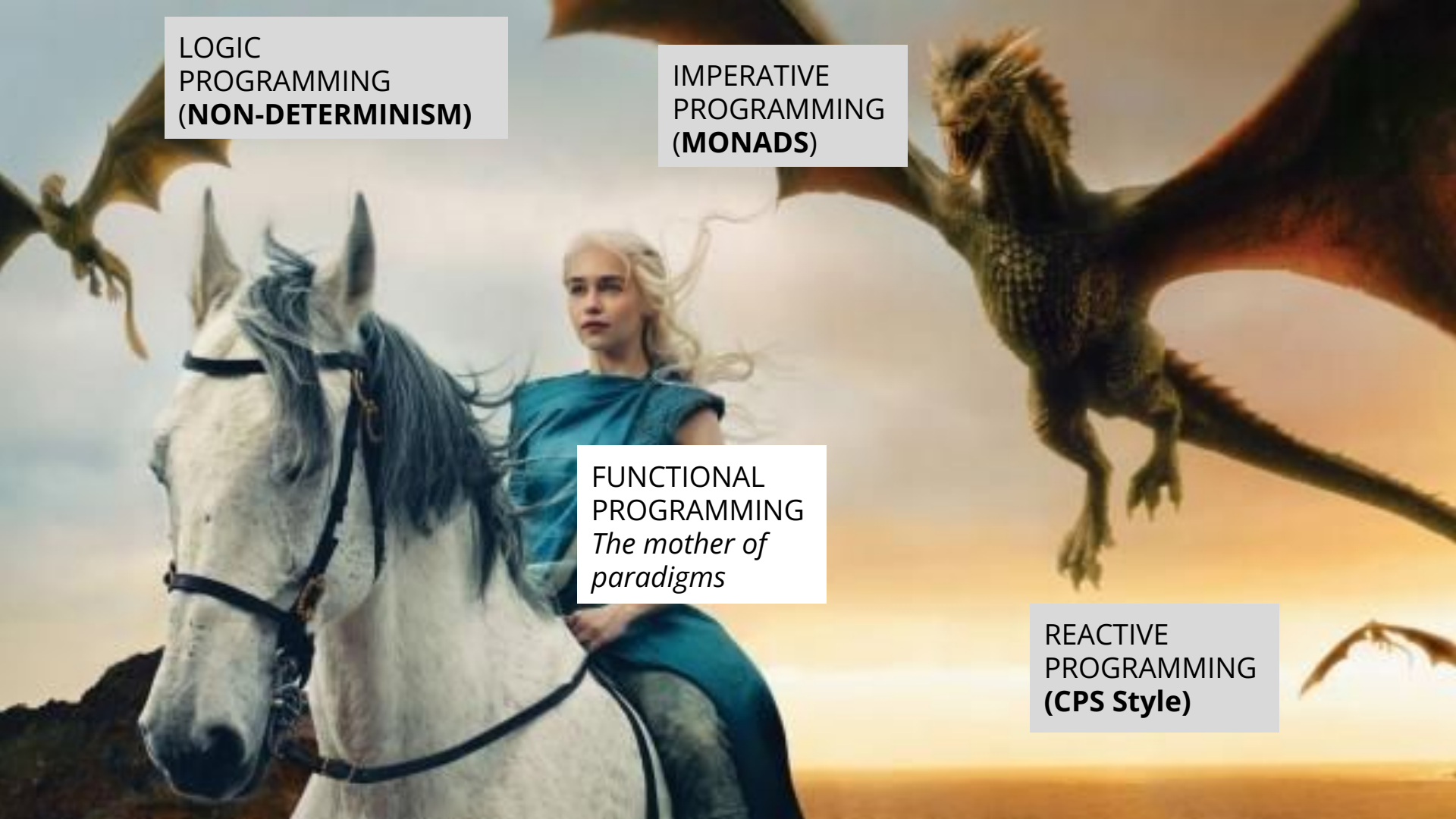
PLANIFICACIÓN PARA EL ESTUDIO DE LA MATERIA PROGRAMACIÓN LÓGICA (28h en aula, 47h fuera de aula)

Sesiones (2h en aula)	Contenidos y objetivos cubiertos	Materiales (con ejercicios)	Prácticas (con ordenador)	Trabajo extra (fuera de aula)
1	Introducción a la Programación Lógica Introducción a PROLOG y a SWISH	PL1 PL2-1	- PL2-1 pg.10	0.5 h 0.5 h
2	Sintaxis de PROLOG	PL2-2	Práctica1, Ej. 1 y 2	2h
3	Semántica de de PROLOG: Unificación	PL2-3, 1 y 2	Práctica1, Ej. 3	3h
4	Semántica de de PROLOG: Regla y Árboles de Resolución	PL2-3, 3 y 4		3h
5	Repaso de sintaxis y semántica de PROLOG	PL2-2 y PL2-3	Práctica1	3h
6	Clasificación y comparación de términos Aritmética Entrada/salida	PL2-4 PL2-5 PL2-6	Práctica2	3h
7	Listas: representación, patrones + "pertenece"	PL2-7, 1	Práctica3, Ej. 1	4h
8	Listas: predicados básicos + ejemplos adicionales	PL2-7, 2 y 3	Práctica3 Ej. 2 (sin corte) y 3	4h
9	Repaso de manejo de listas en PROLOG	PL2-7	Práctica3	4h
10	El predicado de corte: definición, efectos, propiedades	PL2-8, 1	Práctica3, Ej. 2 (ahora con corte)	4h
11	El predicado de corte: usos para estructuras condicionales El predicado de negación	PL2-8, 2 PL3-1	Práctica3, Ej. 3 (ahora con corte)	2h 2h
12	Recolección de soluciones Orden superior: motivación, definición, predicados básicos	PL3-2 PL3-3, 1 y 2	Práctica4, Ej.1.1, 2.1, 2.2 y 2.3 -	1h 1h
13	Predicados de orden superior clásicos (aplicación y filtrado)	PL3-3, 3	Práctica4, Ej.1.2, 2.4	5h
14	Predicados de orden superior: plegado + implementación	PL3-3, 3 y 4	Práctica4, Ej. 3, 4 y 5.	5h

Universidad
Rey Juan Carlos

Programación funcional

Programación declarativa
Grado en Ingeniería Informática
Universidad Rey Juan Carlos


LOGIC PROGRAMMING (**NON-DETERMINISM)**
IMPERATIVE PROGRAMMING (**MONADS**)
FUNCTIONAL PROGRAMMING *The mother of paradigms*
REACTIVE PROGRAMMING **(CPS Style)**

Landmarks in functional programming

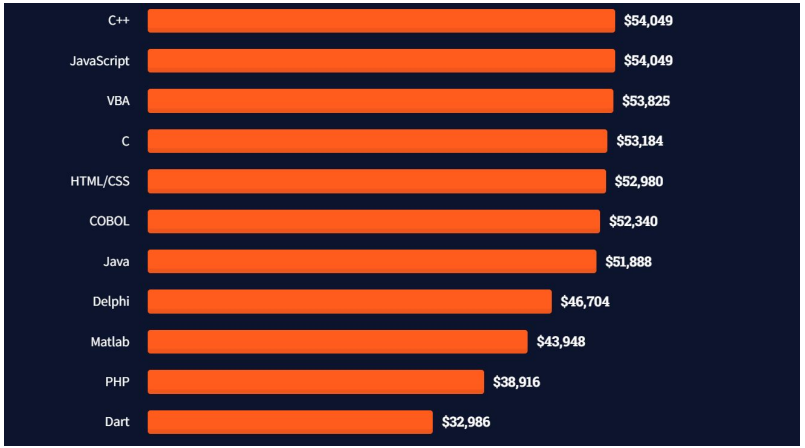
- 1930s- Lambda calculus (Church)
- 1958- LISP (McCarthy)
- 1970s- ML (Milner), HOPE
- 1986- Erlang
- 1987- Haskell
- 1990- Monads in Haskell (Wadler)
- **2004- Scala (Odersky)**
- 2005- F# (Don Syme)
- 2007- Clojure (Hickey)
- *2009- Akka*
- *2010 - Spark 0.1*
- 2014- Java8, Swift (Apple)
- **2021- Scala 3**

¿Por qué Scala?

stackoverflow

https://insights.stackoverflow.com/survey/2021

.....



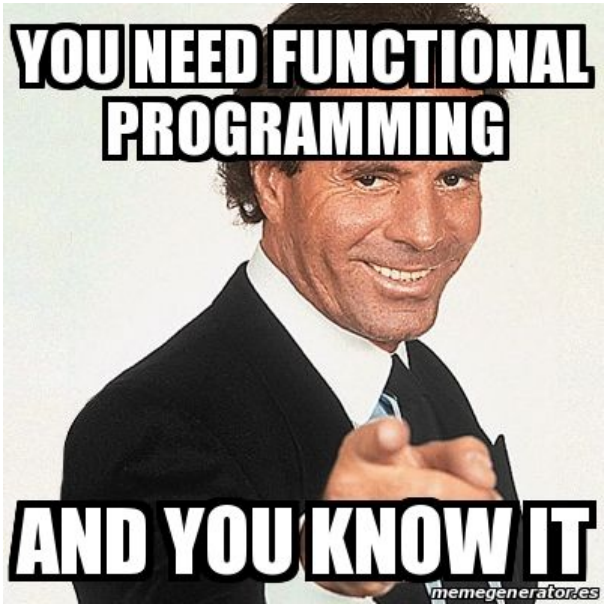
¿Por qué Scala?

Flink

¿Por qué la programación funcional?

Si quieres que tus programas sean fácilmente

- Comprensibles
- Testables
- Mantenibles
- Reutilizables
- Modificables
- Optimizables
- ...



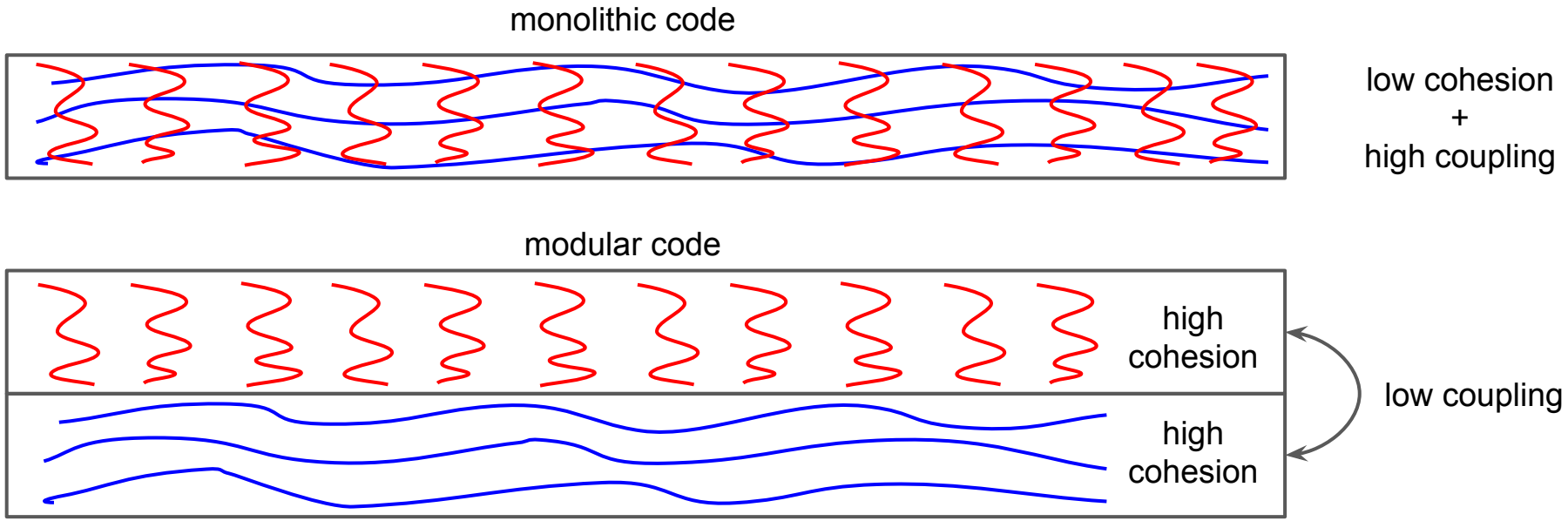
¿Cómo consigue la programación funcional satisfacer estos requisitos no-funcionales?

- Modularity FTW!
  - functions
  - parametric polymorphism
  - higher-order functions
  - Type classes (ad-hoc polymorphism)
  - Languages (domain-specific languages)
  - datatype generics
  - lazy evaluation
  - ...

¿Qué es la modularidad?

- Código monolítico
  - Diferentes conceptos entre-mezclados
  - Difícil de entender, probar, reutilizar, mantener, etc.
- Código modular
  - Cada aspecto del código se encuentra paquetizado en diferentes módulos
  - Fácilmente comprensible, testable, reutilizable, etc.

Modularidad: ¡alta cohesión y bajo acoplamiento!



La escalera de la **modularidad**

DATA-TYPE GENERICS

LANGUAGES

TYPE CLASSES

HIGHER-ORDER FUNCTIONS

PARAMETRIC POLYMORPHISM

FUNCTIONS

PF 3

PF 2

Temario

PF-1 Introducción a la programación funcional

- PF-1.1 El lenguaje Scala

PF-2 Funciones y tipos de datos

- PF-2.1 Funciones y tipos algebraicos de datos
- PF-2.2 Funciones recursivas
- PF-2.3 El isomorfismo Curry-Howard

PF-3 Funciones de orden superior

- PF-3.1 Funciones de orden superior (HOFs)
- *PF-3.2 HOFs como lenguaje de queries*

Planificación (Vicálvaro)

Septiembre

L	M	X	J	V	S	D
			1	2	3	4
5	6	7	8	9	10	11
S 12	13	F 14	15	16	17	18
19	20	F 21	22	23	24	25
ADT 26	27	ADT* 28	29	30		

Octubre

L	M	X	J	V	S	D
					1	2
RF 3	4	RF* 5	6	7	8	9
CH 10	11	12	13	14	15	16
CH* 17	18	HOF 19	20	21	22	23
HOF 24	25	HOF* 26	27	28	29	30
EXAMEN 31						

Planificación (Móstoles)

Septiembre

L	M	X	J	V	S	D
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	S 16	17	18
19	20	21	22	F 23	24	25
F 26	27	28	29	ADT 30		

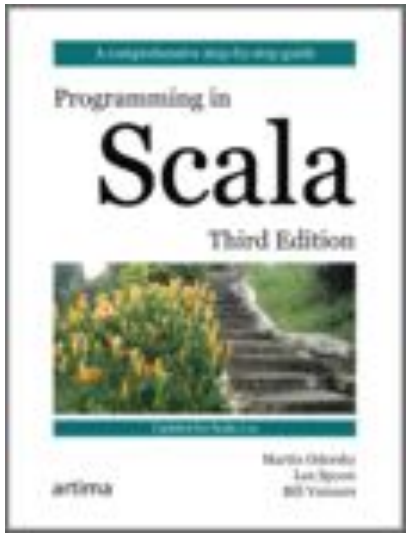
Octubre

L	M	X	J	V	S	D
					1	2
ADT* 3	4	5	6	RF 7	8	9
RF* 10	11	12	13	CH 14	15	16
CH* 17	18	19	20	HOF 21	22	23
HOF 24	25	26	27	HOF* 28	29	30
EXAMEN 31						

Bibliografía

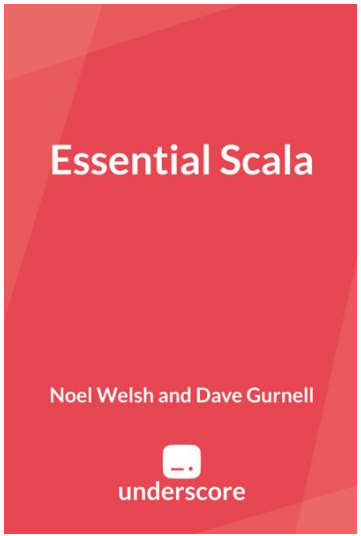
Programming in Scala

M. Odersky, L. Spoons, B. Venners



Essential Scala

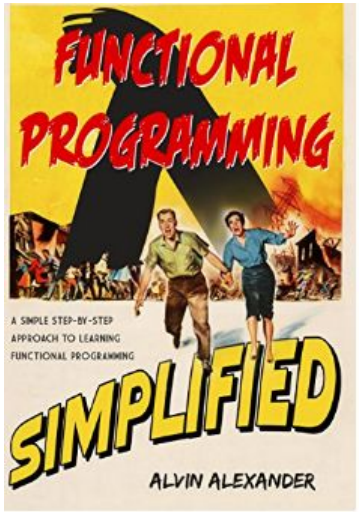
Noel Welsh, Dave Gurnell



Bibliografía

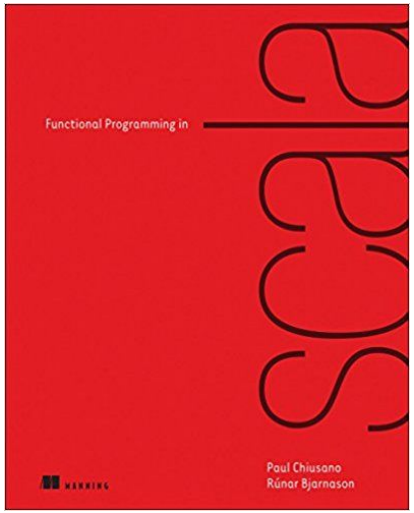
Functional Programming, Simplified

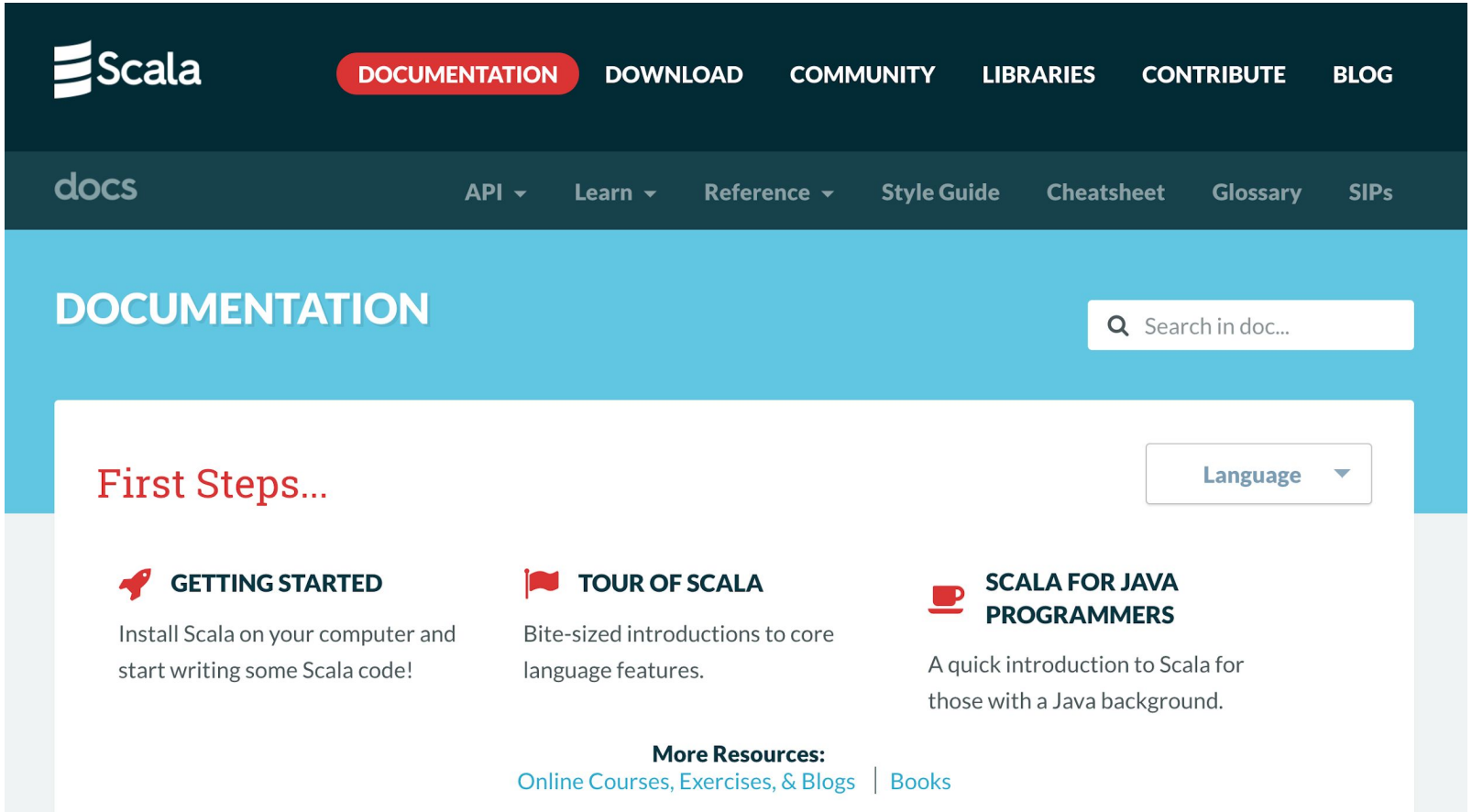
Alvin Alexander



Functional programming in Scala

Chiusano, Bjarnason





http://www.scala-lang.org/documentation/

https://www.meetup.com/Scala-Programming-Madrid/

Meetup

Crear un nuev

ScalaMAD: Scala Programming @ Madrid

Madrid, España

2287 miembros · Grupo público

Organizado por **Juan Manuel S.** y otras 5 personas

Compartir:

Sobre nosotros | Eventos | Miembros | Fotos | Conversaciones | Unirse a este grupo | ...

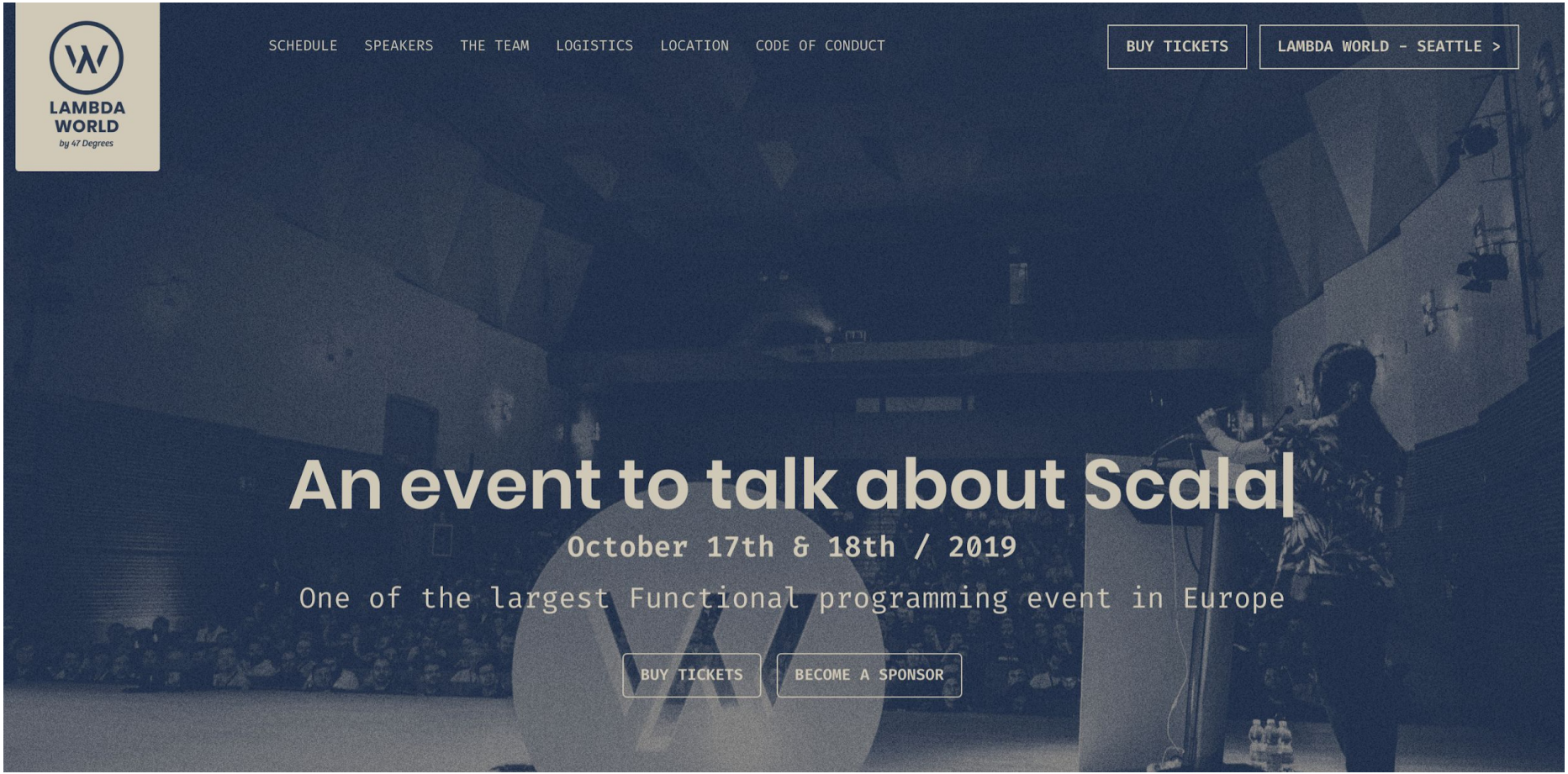
Lo que hacemos

Scala es un lenguaje de programación orientado a objetos y, a la vez, un lenguaje funcional. La combinación de estos dos paradigmas hace especialmente atractiva la programación con Scala, y lo convierte en un

Organizadores

Juan Manuel S. y otras 5 personas

Mensaje



http://cadiz.lambda.world/