아카란?
What is Akka?

«탄력적인 분산 실시간 트랜잭션 처리»
«resilient elastic distributed real-time transaction processing»

우리는 올바른 분산, 동시성, 결함 허용, 확장 가능한 어플리케이션을 작성하는 것은 매우 어렵다고 믿는다.
We believe that writing correct distributed, concurrent, fault-tolerant and scalable applications is too hard.
우리는 잘못된 도구와 잘못된 추상화 수준을 사용하는 것이 대부분의 경우이기 때문이다.
Most of the time it's because we are using the wrong tools and the wrong level of abstraction.
아카는 그것을 바꾸기 위해 있다.
Akka is here to change that.
액터 모델을 사용하여 우리는 추상화 수준을 높이고 확장성, 탄력성, 응답성이 뛰어난 어플리케이션을 빌드할 수 있는 더 나은 플랫폼들을 제공한다—더 자세한 내용을 위해 리액티브 선언문을 참조해라.
Using the Actor Model we raise the abstraction level and provide a better platform to build scalable, resilient and responsive applications—see the Reactive Manifesto for more details.
결함 허용을 위해 우리는 텔레콤 산업이 자가치유가 가능하고 절대 멈추지 않는 시스템 어플리케이션을 빌드하는데 큰 성공을 거두었던 “고장나도록 내버려두다” 모델을 채택했다.
For fault-tolerance we adopt the "let it crash" model which the telecom industry has used with great success to build applications that self-heal and systems that never stop.
액터는 또한 투명한 분산을 위한 추상화와 진정으로 확장이 가능하고 결함 허용 어플리케이션 기반을 제공한다.
Actors also provide the abstraction for transparent distribution and the basis for truly scalable and fault-tolerant applications.

아카는 오픈소스이며 아파치 2 라이센스 안에서 사용이 가능하다.
Akka is Open Source and available under the Apache 2 License.

Download from http://akka.io/downloads.

모든 코드는 컴파일 되어있다. 만약 당신이 직접 그 소스에 접근하길 원한다면 github 에 있는 아카 문서 하위 프로젝트를 살펴봐라.
Please note that all code samples compile, so if you want direct access to the sources, have a look over at the Akka Docs subproject on github: for Java and Scala.

아카는 unique hybrid 를 구현한다
Akka implements a unique hybrid

액터
Actors

액터가 당신에게 주는 것:
Actors give you:

- 배포, 동시성, 병렬처리를 위한 단순하고 높은 수준의 추상화.
  Simple and high-level abstractions for distribution, concurrency and parallelism.
- 비동기, 넌블로킹, 고성능 메세지 중심 프로그래밍 모델.
  Asynchronous, non-blocking and highly performant message-driven programming model.
- 매우 가벼운 이벤트 중심 처리 (GB 힙 메모리 당 수백만 개 액터).
  Very lightweight event-driven processes (several million actors per GB of heap memory).

Scala 또는 *Java* 장을 참조.
See the chapter for *Scala* or *Java*.

결함 허용
Fault Tolerance

- “고장나도록 두다” 라는 의미를 가진 관리자 계층 구조.
  Supervisor hierarchies with "let-it-crash" semantics.
- 액터 시스템은 진정한 결함 허용 시스템을 제공하기 위해 여러개의 JVM 으로 확장할 수 있다.
  Actor systems can span over multiple JVMs to provide truly fault-tolerant systems.
- 자가치유와 절대 멈추지 않는 결함 허용 시스템을 작성하는 것에 우수하다.
  Excellent for writing highly fault-tolerant systems that self-heal and never stop.

결함 허용 (Scala) 와 *결함 허용 (Java)* 을 보라.
See *Fault Tolerance (Scala)* and *Fault Tolerance (Java)*.

위치 투명성
Location Transparency

아카의 모든것은 분산 환경에서 일하도록 설계되어 있다: 모든 액터의 상호작용은 순수한 메세지 전달을 사용하며 모든것은 비동기이다.
Everything in Akka is designed to work in a distributed environment: all interactions of actors use pure message passing and everything is asynchronous.

클러스터 지원에 대한 미리보기는 *Java* 와 *Scala* 설명서를 참조하라.
For an overview of the cluster support see the *Java* and *Scala* documentation chapters.

지속성
Persistence

액터가 경험한 상태 변화는 액터가 시작 또는 재시작 됐을 때 선택적으로 지속되고 재생성 될 수 있다.
State changes experienced by an actor can optionally be persisted and replayed when the actor is started or restarted.
JVM 이 충돌하거나 다른 노드로 마이그레이션 됐을 때에도 액터는 상태를 복구하는 것을 허락한다.
This allows actors to recover their state, even after JVM crashes or when being migrated to another node.

더 자세한 사항은 자바 또는 *스칼라* 리액티브 장에서 찾을 수 있다.
You can find more details in the respective chapter for *Java* or *Scala*.

스칼라와 자바 API
Scala and Java APIs

아카는 스칼라 문서와 자바 문서가 모두 있다.
Akka has both a *Scala Documentation* and a *Java Documentation*.

아카는 다른 방법으로 사용할 수 있다
Akka can be used in different ways

아카는 프레임워크가 아니라 툴킷이다: 특정 소스 코드 레이아웃을 따르지 않은채 다른 라이브러리처럼 당신의 빌드로 구성 할 수 있다.
Akka is a toolkit, not a framework: you integrate it into your build like any other library without having to follow a particular source code layout.

When expressing your systems as collaborating Actors you may feel pushed more towards proper encapsulation of internal state, you may find that there is a natural separation between business logic and inter-component communication.

아카 어플리케이션은 일반적으로 다음과 같이 배포된다.
Akka applications are typically deployed as follows:

- 라이브러리로: 클래스패스 또는 웹 어플리케이션에서 일반적인 JAR 로 사용된다.
  as a library: used as a regular JAR on the classpath or in a web app.
- sbt-native-packager 로 패키징 된다.
  packaged with sbt-native-packager.
- Lightbend ConductR 을 사용하여 패키징 되고 배포된다.
  packaged and deployed using Lightbend ConductR.

상업 지원
Commercial Support

아카는 Lightbend Inc 에서 구할 수 있다.
Akka is available from Lightbend Inc.
개발 또는 제품 지원을 포함하는 상업 라이센스에는, 여기를 더 읽어라.
under a commercial license which includes development or production support, read more here.