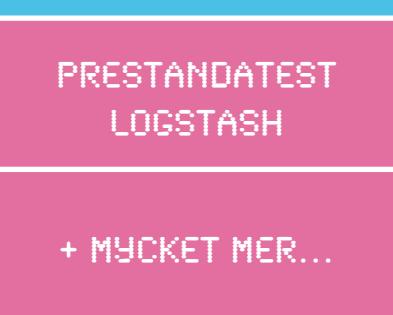
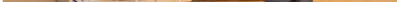
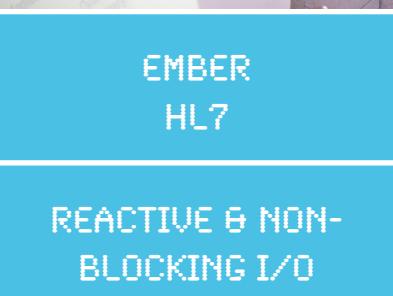
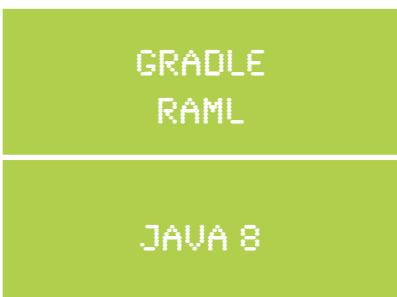


# CADEC 2015

CALLISTA DEVELOPER'S CONFERENCE DEN 28 JANUARI 2015



## VI BJUDER PÅ DE SENASTE TRENDERNA INOM IT-ARKITEKTUR SAMT AGIL & ÖPPEN SYSTEMUTVECKLING

Även 2015 kommer att starta med en rykande aktuell Cadec som kommer att hållas på IHM i Gårda, Göteborg, precis som tidigare. Konferensen är kostnadsfri.

Förmiddagen består av tre parallella hands-on tutorials: Ember, Angular-dödaren på Frontendsidan, Skalbara serverapplikationer med non-blocking I/O och reactive programming och en favorit i repris från förra året Testautomation med dynamiska språk. Eftermiddagen fylls som vanligt med ett gediget konferensprogram som ni hittar nedan. Missa heller inte att varva ner efter konferensen med mat, dryck och mingel på After Cadec.

Varmt välkommen till en heldag med kompetensutveckling!

Detaljerad information och anmälan på vår hemsida

[callistaenterprise.se/cadec2015/](http://callistaenterprise.se/cadec2015/)



## FRÅN MAVEN TILL GRADLE

JESPER HOLMBERG

Sedan några år tillbaka är Gradle ett modernt och spänande alternativ till äldre byggsverktyg som Ant och Maven. Gradle lånar många idéer från sina föregångare, men erbjuder användaren en helt ny flexibilitet och uttrycksfullhet. Ett smidigare och mer kraftfullt byggsverktyg ger många fördelar när projekten växer och blir mer heterogena, men även för mindre projekt är Gradle ett utmärkt verktyg för att skapa kompakta, läsbara och lättföränderliga bygglösningar.

Från Maven ärver Gradle idén om "konvention över konfiguration", vilket leder till att många inställningar inte behöver specificeras alls om man följer de föreslagna konventionerna. Dessutom har Gradle ärvit många konventioner från Maven, vilket gör att den erfarne Maven-användaren kan använda många av sina invanda mönster i Gradle-världen.

Men trots likheterna och den gemensamma historien är det många saker som skiljer Maven och Gradle. Vi kommer i detta föredrag att titta på en del skillnader mellan byggsverktygen, och hur man förhåller sig till dessa skillnader om man vill ersätta Maven med Gradle i ett existerande projekt.

Vi kommer att gå igenom en del kända fallgropar, men också diskutera de skillnader i grundidéer mellan verktygen som gör att man till viss del behöver förändra sitt sätt att tänka kring existerande lösningar för att på bästa sätt dra nytta av Gradles fördelar.

## LOGGÖVERVAKNING I REALTID MED ELK

OLA DEIBITSCH, HANS THUNBERG

Hur kan vi säkerställa att applikationer fungerar? Hur kan vi ge oss möjlighet att ligga steget före för att ge utrymme för att arbeta proaktivt?

Loggövervakning i realtid ger inte bara möjlighet att enkelt fånga eventuella fel, utan ger även möjlighet att analysera trender och tendenser.

I detta blixtföredrag så kommer Hans och Ola titta närmare på och utforska Elasticsearch, Logstash och Kibana (ELK). Tre sömlöst integrerade open source-produkter, vilka gör det enklare än någonsin att centralisera, söka, analysera och visualisera logghändelser i realtid.



## REACTIVE

MAGNUS LARSSON

Med ett ständigt ökande antal uppkopplade enheter (mobilier, plattor, "Internet of things" ...) ökar kravet på att kunna hantera många samtidiga anrop till våra system. Traditionellt har Java baserade system använt så kallad "blocking I/O", dvs allokerat en tråd för varje inkommende anrop. Detta resulterar i begränsad skalbarhet samt dålig tolerans för fel orsakat primärt av dränerade trådpooler.

En nyckelfaktor för att lösa dessa problem är att utnyttja asynkrona mekanismer i underliggande operativsystem som ger möjlighet till så kallad "non-blocking I/O". Denna teknik allokerar inte en tråd per anrop vilket leder till betydligt bättre skalbarhet och feitolertans.

Användning av "non-blocking I/O" leder dock till en callback baserad programmeringsmodell. I lite mer komplexa fall leder detta till svårhanterlig källkod, också känt som "callback hell".

För att råda bot på "callback hell" har det vuxit fram ett antal ramverk som är baserade på en ansats som kallas "reactive programming".

Föredraget kommer att belysa dessa problem och möjligheter med en blandning av teori, demonstrationer och kodexempel.

## FHIR OCH INTEROPERABILITET I SJUKVÅRDEN

OSKAR THUNMAN

Hälso- och sjukvården hör till de branscher där rätt information vid rätt tillfälle kan vara en fråga om liv och död. Ändå är digitaliseringen av hälso- och sjukvården eftersatt och förutsättningarna för utbyte av information är dåliga. Sedan år 2001 har standardisering av meddelanden (XML via SOAP) varit det verktyg man använt för att uppnå interoperabilitet. De standardmeddelanden man tagit fram har varit kostsamma och endast kunnat uppnå en minsta gemensamma nämnare mellan de inblandade systemen.

2012 kläcktes idén att tackla interoperabiliteten med hjälp av REST-API:er genom en ny ansats till standard kallad FHIR (Fast Healthcare Interoperability Resource). Detta föredrag beskriver hur pass revolutionerande denna standard faktist är, varför "alla" älskar den och hur REST kan bidra till innovation inom hälso- och sjukvård.

KONFERENS  
13.00 - 17.30



## RAML

OLA DEIBITSCH

Under Cadec 2014 frågade vi oss huruvida RAML, som ett nytt initiativ för att beskriva och uppmuntra till bättre API:er, skulle få momentum. Ett år har gått, men vad har hänt sedan dess? I detta blixtföredrag tittar vi inte bara närmare nyheterna i RAML 1.0, utan även snabbt vad som hänt med verktygslådan.

## JAVA 8 - NU I PRAKTIKEN

ROGER LINDSJÖ

Med Java 7 EOL runt hörnet gör att vi snart måste uppgradera till Java 8. På vilket sätt kan vi då passa på att dra fördelar av de nya funktioner som erbjuds?

Är lambdauttryck och strömmar bara för funktionella språk eller kan vi dra nytta av det i "smör och bröd" kod. Hur svårt kan det vara med datum? Är det värt besväret att använda java.time.\*? Jag använder ju redan Joda time? Exempel på kod som jag stöter på i dagligt arbete som kan bli enklare och tydligare med Java 8.

## EMBER

STEPHEN WHITE

Ember "is a framework for creating ambitious web applications" not a framework for creating ambitious frameworks ...

In this presentation Stephen will show how understanding the MVC pattern as realized by Ember can lead to a productive, structured, testable and maintainable code base.

Understanding MVC isn't that tough and with the use of ember-cli one can get up-to speed fast. With embers embrace of ECM6 one can create structured modular code in line for the future of javascript and the web.

KONFERENS  
13.00 - 17.30



## PRESTANDATESTNING I AGILA PROJEKT

BJÖRN BESKOW

Xtreme Programming satte automatiserad enhetstestning på kartan för den moderna utvecklaren. Test-driven Utveckling (TDD) lyfter värdet av tidig test-automation till sin spets. ATDD gör detsamma för acceptansteller. Mognadsgraden i branschen kring tidig funktionell kvalitets-säkring via automatiserade tester är stor, och utgör en viktig möjliggörare för framgångsrik agil utveckling.

Men när det kommer till kvalitets-säkring av icke-funktionella aspekter så som prestanda, skalbarhet och last-tålighet är dock bilden en helt annan. Icke-funktionell testning betraktas fortfarande som svårt. Verktygen har traditionellt varit dyra och komplicerade, och ofta ställt höga krav på både kompetens, licenser och infrastruktur. Därmed faller ofta kvalitetssäkring av icke-funktionella aspekter utanför många projekts definition av "Done", och blir något som skall göras "senare". I bästa fall görs de inför första leverans, om ens någonsin. Detta är paradoxalt, då eventuella problem med icke-funktionella krav ofta får långtgående konsekvenser.

En av de stora utmaningarna vid prestanda- och last-testning är generering av adekvat last. Det är ofta detta som gjort traditionella verktyg dyra och komplexa. Men en ny generation verktyg för prestanda-testning tar icke-blockerande IO till hjälp för att rita om spelplanen.

Detta föredrag går helt kort igenom en ansats för tidig automation av prestanda- och last-tester, exemplifierat med Gatling (<http://gatling.io>).

HANDS-ON  
TUTORIALS  
9.00 - 12.00



Cadec Tutorials är till för dig som är nyfiken på nya teknikområden och vill få möjlighet att testa "hands-on". Övningarna sker på din egen medhavda laptop. Instruktioner och nedladdningsbart paket med utvecklingsmiljö etc. kommer att finnas tillgängligt på vår hemsida några dagar i förväg. En tutorial varar hela förmiddagen.

## TESTAUTOMATION MED DYNAMISKA SPRÅK

BJÖRN BESKOW

Effektiv testautomation stupar ofta på tekniska tillkortakommanden i verktyg och ramverk. Istället för att kunna koncentrera sig på de verkliga utmaningarna inom test, tvingas vi ofta lägga orimligt mycket tid på teknikaliteter kring själva automationen. Här är inte alltid ett traditionellt, starkt typat, språk som Java helt ändamålsenligt, även om den produktionskod som skall testas är Java-baserad.

I denna tutorial tittar vi på hur ett modernt, dynamiskt typat språk som Groovy kan ge avsevärt enklare, effektivare och mer läsbara lösningar på de vanligaste utmaningarna inom testautomation. Vi täcker in ett brett spektrum – från enhetstestning, integrationstestning och api-testning till acceptanstestning och gui-automation.

Vi bygger vår tutorial på interaktiva övningar blandat med korta teori-genomgångar. Deltagare förutsätts ha viss kunskap och erfarenhet av test-automation på Java-plattformen. Erfarenhet av Groovy eller andra dynamiska språk förutsätts inte.

## EMBER

SEDINA ORUC, STEPHEN WHITE, JONAS BEHMER

I dagens webapplikationer förflyttas allt mer av presentationslogik och renderering av HTML till klienten för att skapa en mer dynamisk och fullödig upplevelse för användaren. Javascript-ramverk som Backbone, Angular och Ember är tre framstående exempel på denna trend.

I denna tutorial får du som funderar på att bygga rika, skalbara och ambitiösa webapplikationer en introduktion i Ember.js och hur dess filosofi med konvention över konfiguration förenklar livet för dig som utvecklare. Teori om ramverkets byggstenar och designprinciper varvas med ett konkret exempel där vi implementerar en applikation. Vi börjar med det grundläggande, dvs. "bootstrapping" av en typisk Ember-applikation, för att därefter bygga upp en modulär applikation som är kapabel att kommunicera med REST-tjänster.

Vi räknar med att du som vill delta har med dig en dator samt grundläggande kunskap i JavaScript, CSS och HTML.

HANDS-ON  
TUTORIALS  
9.00 - 12.00



## SKALBARA SERVERLÖSNINGAR MED NON-BLOCKING I/O OCH REACTIVE PROGRAMMERING

MAGNUS LARSSON, MATS EKHAMMAR

Med ett ständigt ökande antal uppkopplade enheter (mobiler, plattor, "Internet of things" ...) ökar kravet på att kunna hantera många samtidiga anrop till våra system. Traditionellt har Java baserade system använt så kallad "blocking I/O", dvs allokerat en tråd för varje inkommande anrop. Detta resulterar i begränsad skalbarhet samt dålig tolerans för fel orsakat primärt av dränerade trådpooler.

En nyckelfaktor för att lösa dessa problem är att utnyttja asynkrona mekanismer i underliggande operativsystem som ger möjlighet till så kallad "non-blocking I/O". Denna teknik allokerar inte en tråd per anrop vilket leder till betydligt bättre skalbarhet och feitolertans.

Användning av "non-blocking I/O" leder dock till en callback-baserad programmeringsmodell. I lite mer komplexa fall leder detta till svårhanterlig källkod, också känt som "callback hell".

För att råda bot på "callback hell" har det vuxit fram ett antal ramverk som är baserade på en ansats som kallas "reactive programming".

I denna tutorial kommer du att få se prov på ovanstående problem samt möjliga lösningar i ett antal övningar. Vi kommer tillhandahålla en VirtualBox image ([www.virtualbox.org](http://www.virtualbox.org)) som skall användas under övningarna. Imagen är baserad på Linux och innehåller en utvecklingsmiljö med bla Java 8, Git och Eclipse. Grundläggande kunskap om dessa verktyg rekommenderas för att kunna tillgodogöra sig övningarna fullt ut.