Clojure-Web-Applikationen für Beginner

BOB 2016

innoQ

Michael Vitz

Consultant @ innoQ

michael.vitz@innoq.com

@michaelvitz

Clojure

- Lisp-Variante auf JVM
- Funktionale Sprache
- Dynamisch typisiert
- komplettes mächtiges Macrosystem

Datenstrukturen

3 3.14 3/2

"Hello World" \a #"Ch.*se"

foo :first

("Hello", :first)

{ :name "Michael",

:company "innoQ" }

[3 4 3]

#{3 4 3}

Syntax

“You’ve just seen it”

- Rich Hickey

Funktionen

```
(+ 1 2)
› 3
(:city {:name “innoQ”
        :city “Monheim”})
› “Monheim”

(map inc [1 2 3])
› (2 3 4)
```

Funktionen

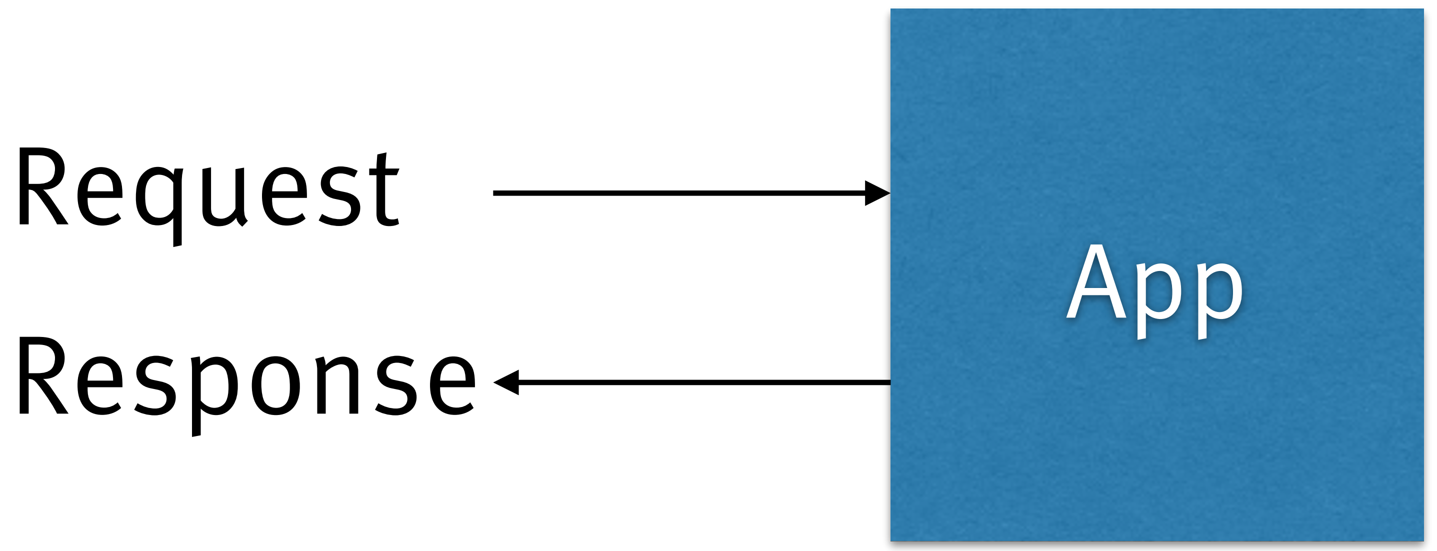
```
(fn [x y] (+ x y))

(def add
  (fn [x y] (+ x y)))

(defn add
  [x y]
  (+ x y))
```

Links

- http://clojure.org
- blog.cognitect.com/blog/2016/1/28/state-of-clojure-2015-survey-results
- https://juxt.pro/radar.html
- http://www.clojure-toolbox.com

Web-Applikationen

Web-Applikation



Web-Applikation

App(Request) = Response

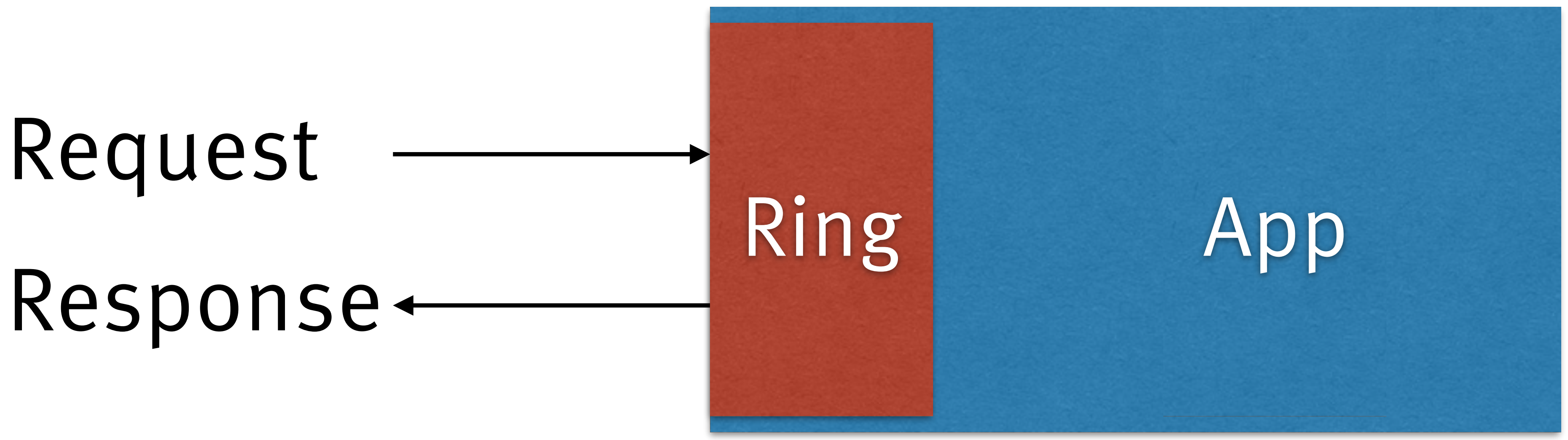
Web-Applikation

```
(defn greet-controller
 [request]
 {:status 200
  :headers {“Content-Type” “text/plain”}
  :body “Hello, world!”})
```

Ring

https://github.com/ring-clojure/ring

Ring



Ring: Handler

```
(defn greet-controller
  [request]
  {:status 200
   :headers {“Content-Type” “text/plain”}
   :body “Hello, world!”})
```

Ring: Middleware

```
(defn wrap-logging
 [handler]
 (fn [request]
  (print request)
  (let [response (handler request)]
   (print response)
   response)))
```

Ring: Middleware

- https://github.com/ring-clojure/ring-defaults
- https://github.com/ring-clojure/ring-json

Ring: Adapter

- Jetty
- Servlet
- http-kit
- Tomcat
- …

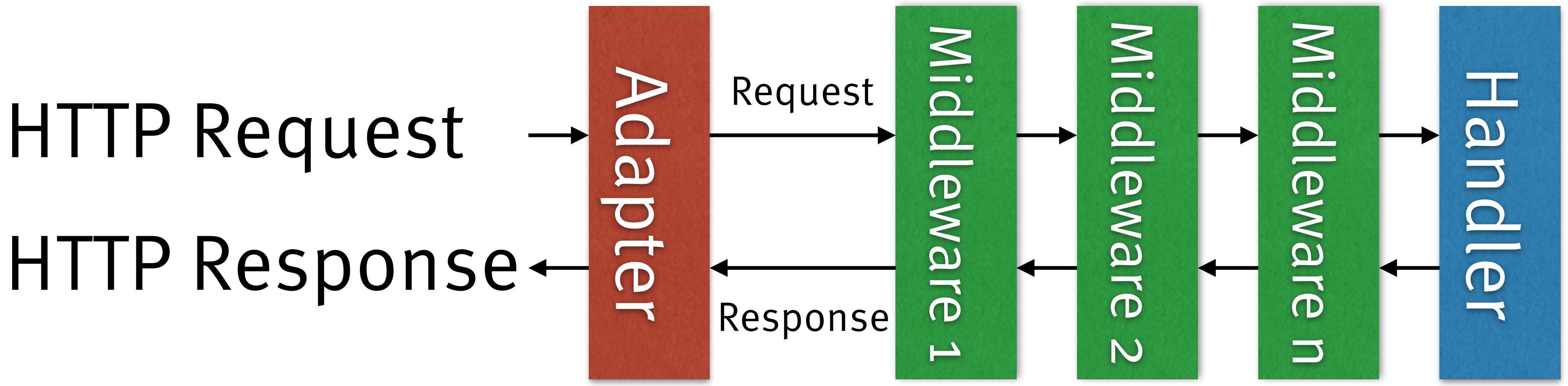
Ring: Request

```
{:uri "/greet"
 :query-string "name=Michael"
 :request-method :get
 :headers { … }
 …}
```

Ring: Response

```
{:status 200
  :headers { … }
  :body “Hello, Michael!”}
```

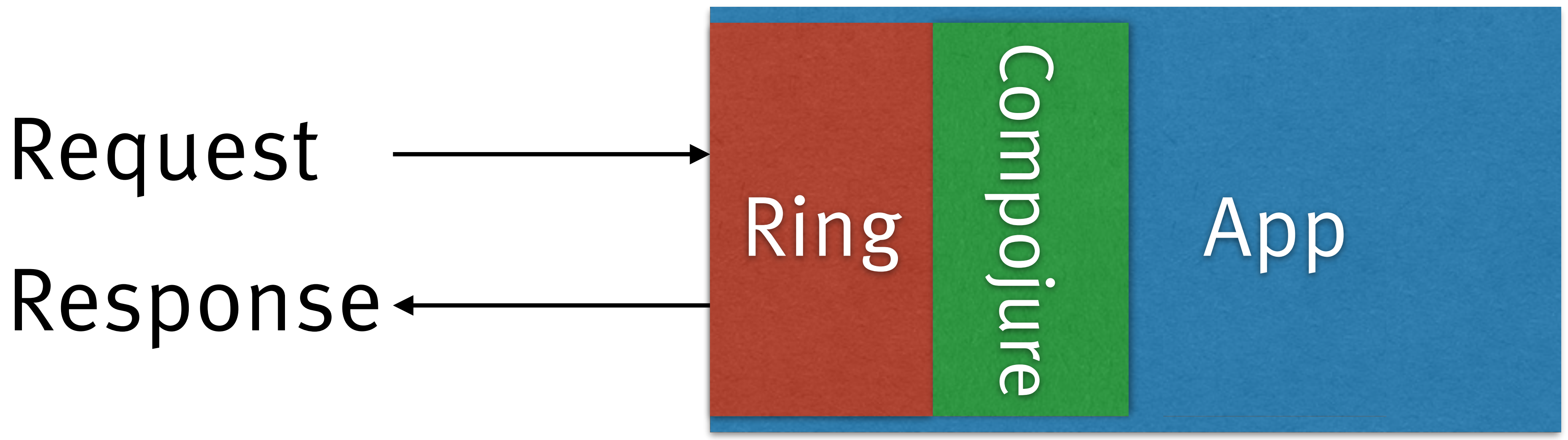
Ring


HTTP Request
HTTP Response
Adapter
Request
Response
Middleware 1
Middleware 2
Middleware n
Handler

Compojure

https://github.com/weavejester/compojure

Compojure



Compojure: Handler

```
(def handler
  (GET “/hello” [] “Hello, world!”))

(handler {:request-method :get, :uri “/hello”})
› {:body “Hello, world!”}

(handler {:request-method :post, :uri “/hello”})
› nil
```

Exkurs: Macros

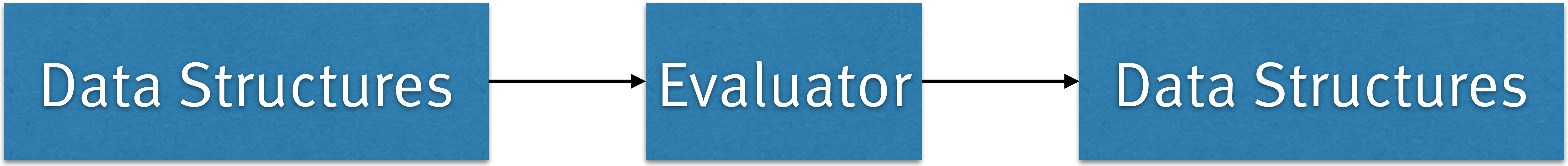
Clojure: Macros

$$\begin{tikzcd}\boxed{\text{Text}} \arrow[r] & \boxed{\text{Reader}} \arrow[r] & \boxed{\text{Data Structures}}\end{tikzcd}$$

```
"(case 3
  1 "one"
  2 "two"
  "more")"
```

```
(case 3
  1 "one"
  2 "two"
  "more")
```

Clojure: Macros



```
(case 3
  1 “one”
  2 “two”
  “more”)
```

```
(if (= 3 1)
  “one”
  (if (= 3 2)
    “two”
    “more”))
```

Zurück zu Compojure…

Compojure: Macro

```
(GET “/hello” [] “Hello, world!”)

(fn [req]
  (if (and (= (:uri req) “/hello”)
               (= (:request-method req) :get))
    {:body “Hello, world!”}))
```

Compojure: Path

```
(GET “/hello/:name” [name]
  (str “Hello, ” name “!”))
```

Compojure: Routes

```
(def my-routes
 (routes
  (GET “/hello” [] “Hello!”)
  (GET “/bye” [] “Bye!”)))

(defroutes my-routes
  (GET “/hello” [] “Hello!”)
  (GET “/bye” [] “Bye!”))
```

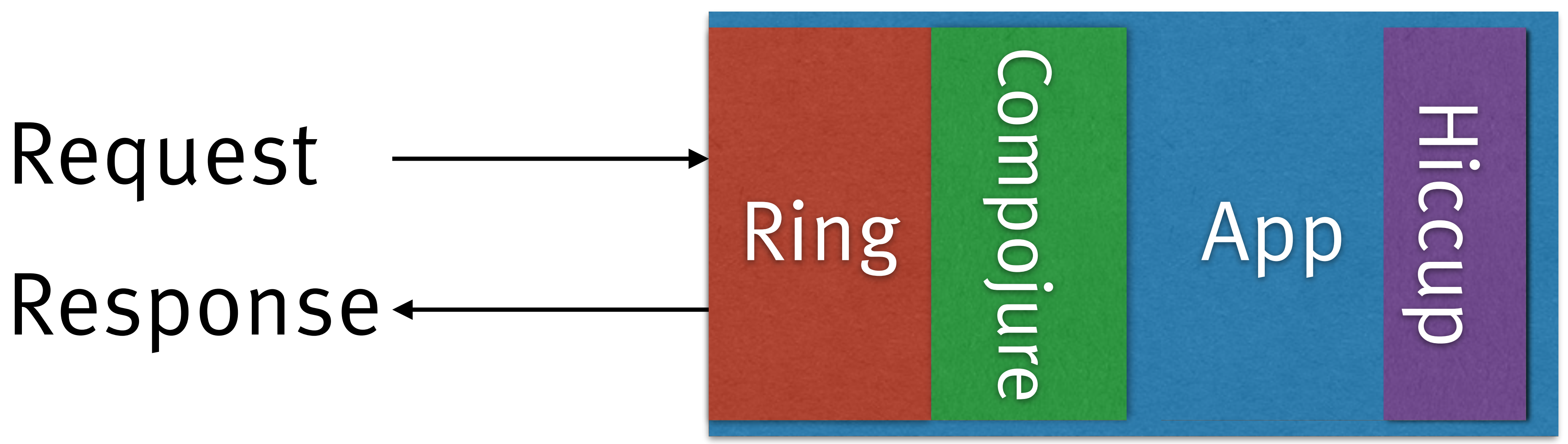
Compojure: Alternativen

- https://github.com/juxt/bidi
- http://clojure-liberator.github.io/liberator/

Hiccup

https://github.com/weavejester/hiccup

Hiccup



Hiccup: Basics

```
[:div {:id “foo”}
  [:span “bar”]]

‹div id=“foo”›
  ‹span›bar‹/span›
‹/div›
```

Hiccup: Shortcuts

```
[:div#foo
  [:span “bar”]]

<div id=“foo”>
  <span>bar</span>
</div>
```

Hiccup: Links

```
(link-to “www.innoq.com” “innoQ”)

[:a {:href “www.innoq.com”} “innoQ”]

<a href=“www.innoq.com”>
  innoQ
</a>
```

Hiccup: Formulare

```
(form-to [:post “/login”]
  (text-field “Username”)
  (password-field “Password”)
  (submit-button “Login”))
```

Hiccup: Alternativen

- https://github.com/danlarkin/clabango
- https://github.com/plakat/clj-thymeleaf
- https://github.com/cgrand/enlive
- https://github.com/yogthos/Selmer

Fazit

Fazit

- Funktionale Sprache ideal für Web-App
- Verbreitung nimmt zu
- Libraries anstelle von Frameworks
- Community gesund und hilfsbereit

Interessante Libraries

- https://github.com/technomancy/leiningen
- https://github.com/weavejester/environ
- https://github.com/clojurewerkz/route-one
- https://github.com/krisajenkins/yesql
- https://github.com/seancorfield/clj-time

Frameworks

- https://github.com/weavejester/duct
- http://www.luminusweb.net
- https://github.com/otto-de/tesla-microservice
- https://github.com/metosin/compojure-api

Danke!

Fragen?

Kommentare?

Michael Vitz | @michaelvitz

michael.vitz@innoq.com

https://www.innoq.com/en/talks/2016/02/clojure-webapps/

innoQ

www.innoq.com

innoQ Deutschland GmbH

Krischerstr. 100	Ohlauer Straße 43	Ludwigstraße 180 E	Kreuzstr. 16
40789 Monheim am Rhein	10999 Berlin	D-63067 Offenbach	D-80331 München
Germany	Germany	Germany	Germany
Phone: +49 2173 3366-0			