Under the Hood …

Ida y vuelta de un request en Seaside

Saber o no saber … esa es la pregunta

- Utilizar un framework implica conocer respetar ciertas reglas y convenciones

- En el caso de Seaside (algunos ejemplos)
  - Cada componente debe ser subclase de WAComponent
  - Los componentes deben implementar el mensaje #renderContentOn:
  - Debo registrar el componente “raíz” en el WAAdmin
  - En los “callbacks” no envío mensajes “al Canvas”

Saber o no saber … esa es la pregunta

- Por lo general no nos hacemos muchas preguntas al respecto (la idea es no tener que aprender lo que no necesito aprender), pero…

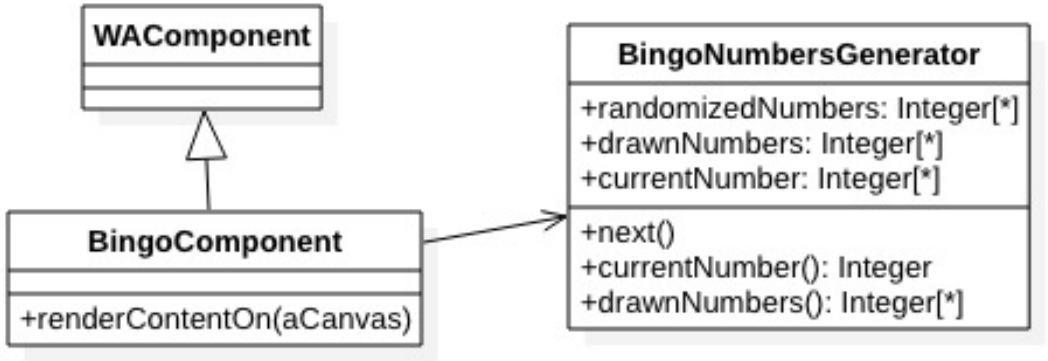
- A veces, conocer como funciona el framework nos ayuda a tomar mejores decisiones y entender cuando las cosas no funcionan
- En OO2, nos interesa aprender investigando buenos diseños de frameworks

Vamos a mirar debajo del capot de Seaside

- Investiguemos:
  - ¿Quién y cuando se crea la instancia de mi componente?
  - ¿Cuantas instancias de mi componente hay en un momento dado?
  - ¿cuándo se envía el mensaje #renderContentOn:?
  - ¿Por qué no hay que enviar mensajes al canvas en los callbacks?
  - … y en el camino tal vez descubrimos alguna otra cosa interesante…

Bingo! (http://localhost:8080/bingo)

- Vamos a usar como ejemplo un simple generador de números de Bingo.
- Tenemos dos clases:
  - El generador de números de Bingo (BingoNumbersGenerator)
  - El componente que muestra los números y permite sacar uno nuevo.



Vamos a mirar debajo del capot de Seaside

- Investiguemos:
  - ¿Quién y cuando se crea la instancia de mi componente?
  - ¿Cuantas instancias de mi componente hay en un momento dado?
  - ¿Cuándo se envía el mensaje #renderContentOn:?
  - ¿Por qué no hay que enviar mensajes al canvas en los callbacks?
  - … y en el camino tal vez descubrimos alguna otra cosa interesante…

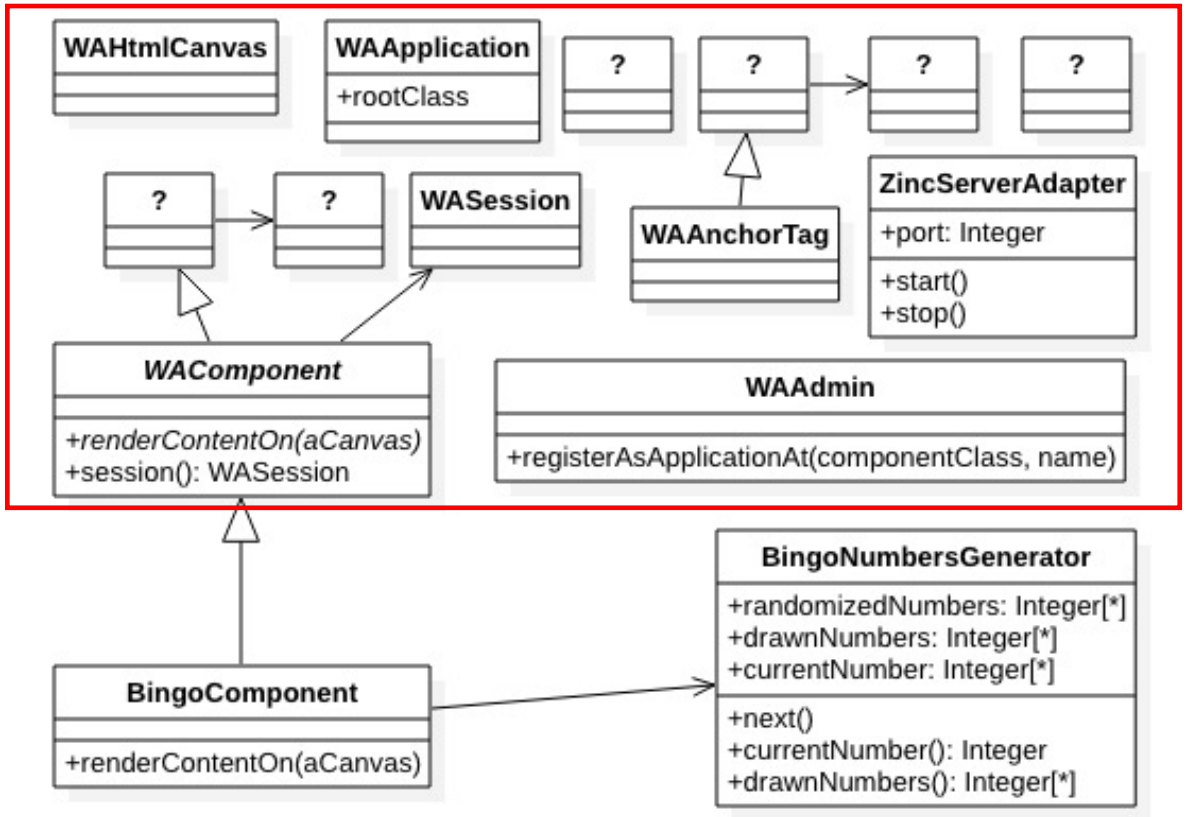
Vamos a mirar debajo del capot de Seaside

- Investiguemos:
  - ¿Quién y cuando se crea la instancia de mi componente?
  - ¿Cuantas instancias de mi componente hay en un momento dado?
  - ¿Cuándo se envía el mensaje #renderContentOn:?
  - ¿Por qué no hay que enviar mensajes al canvas en los callbacks?
  - … y en el camino tal vez descubrimos alguna otra cosa interesante…

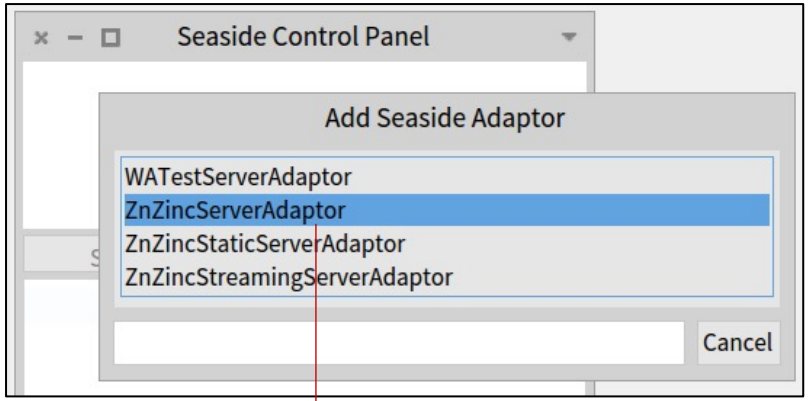
Vamos a mirar debajo del capot de Seaside

- Investiguemos:
  - ¿Quién y cuando se crea la instancia de mi componente?
  - ¿Cuantas instancias de mi componente hay en un momento dado?
  - ¿Cuándo se envía el mensaje #renderContentOn:?
  - ¿Por qué no hay que enviar mensajes al canvas en los callbacks?
  - … y en el camino tal vez descubrimos alguna otra cosa interesante…

Clases del Framework

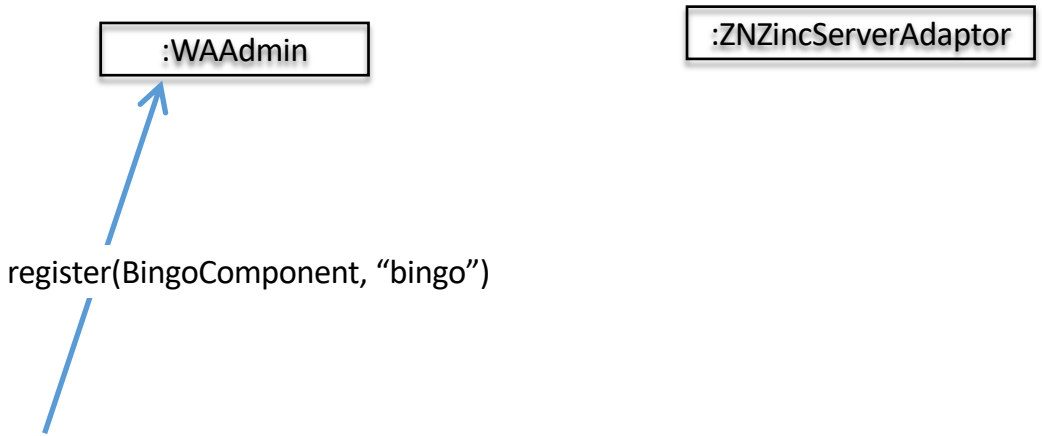


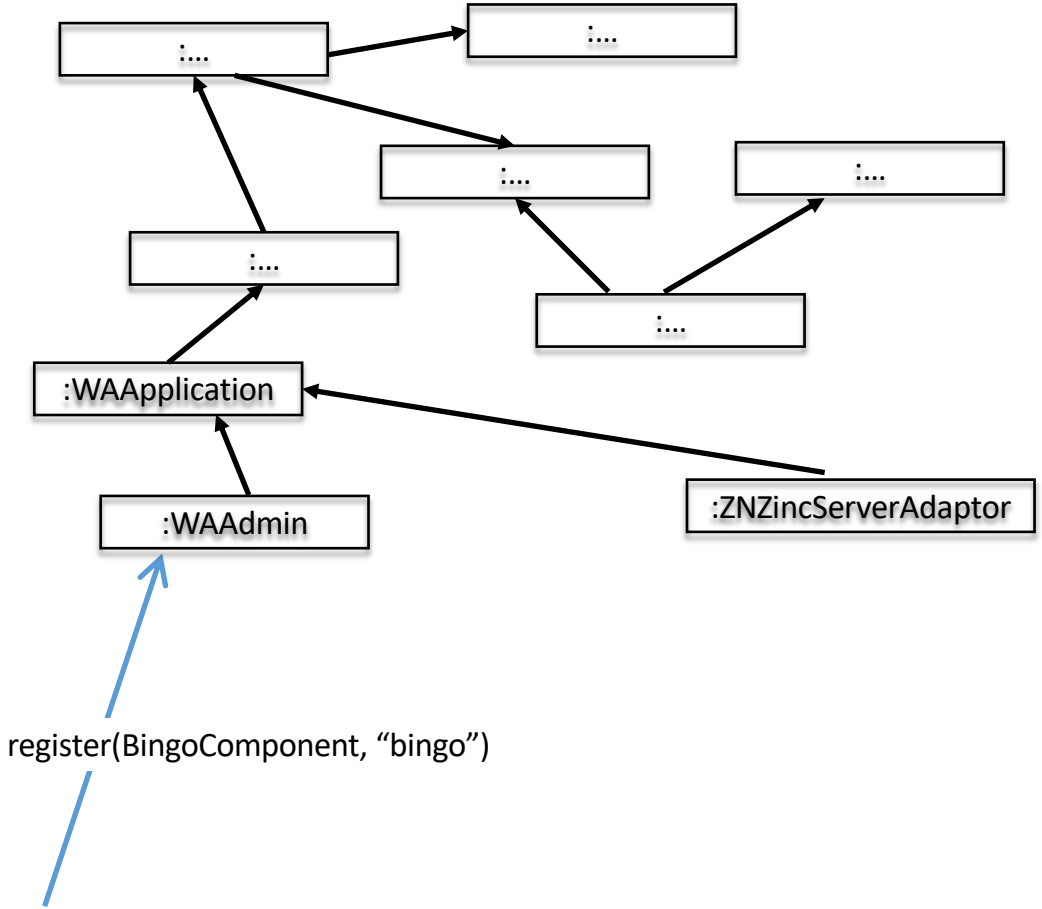
Clases de mi Aplicación

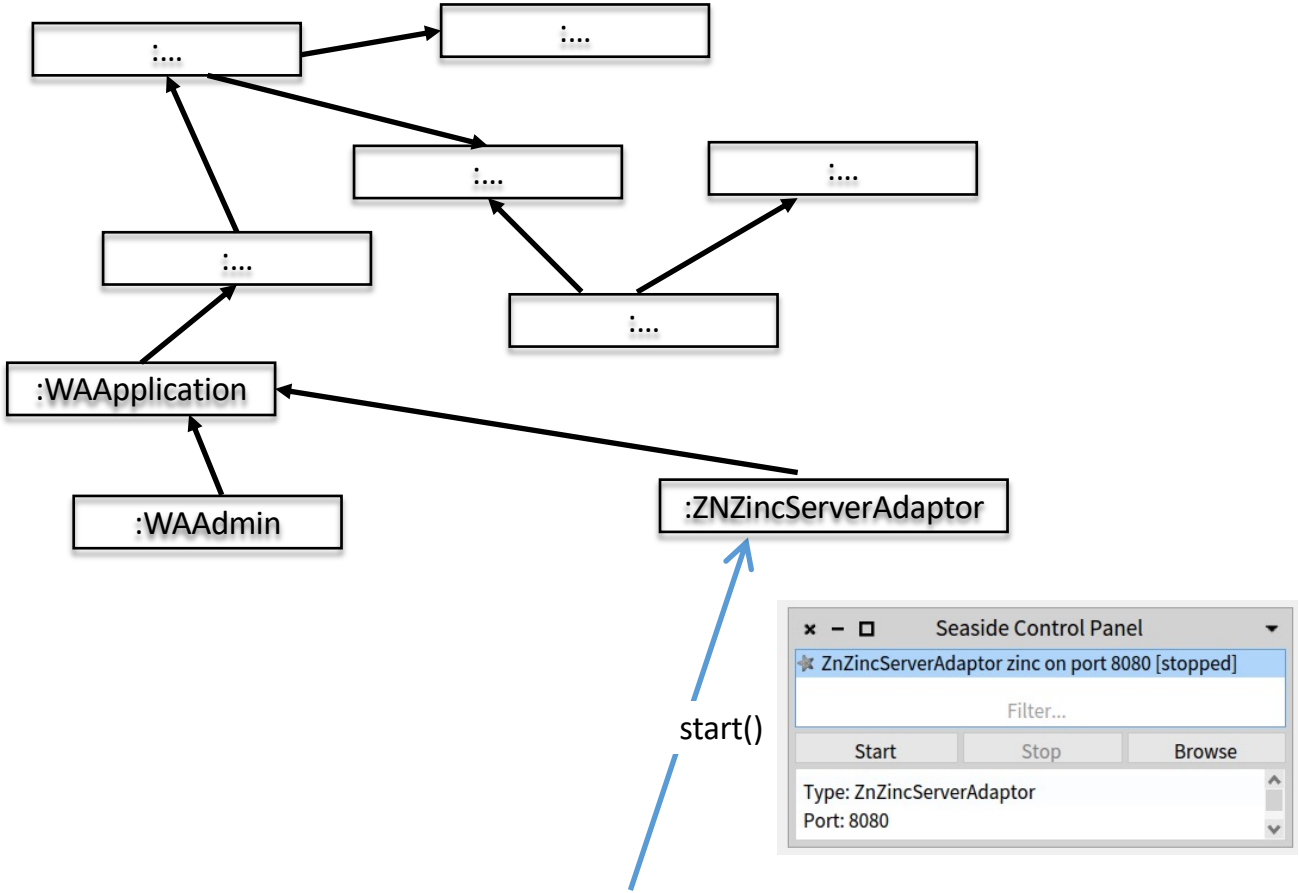


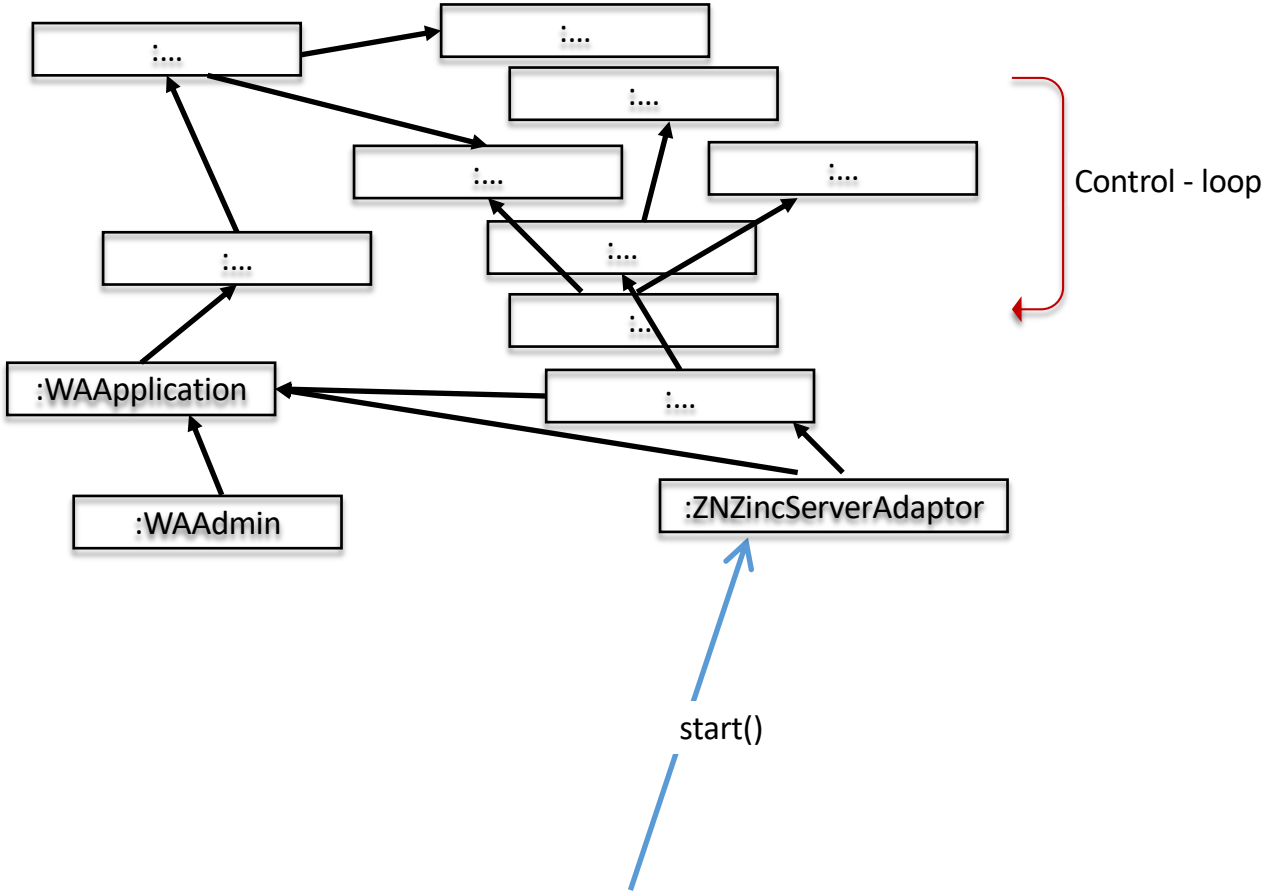
:WAAdmin

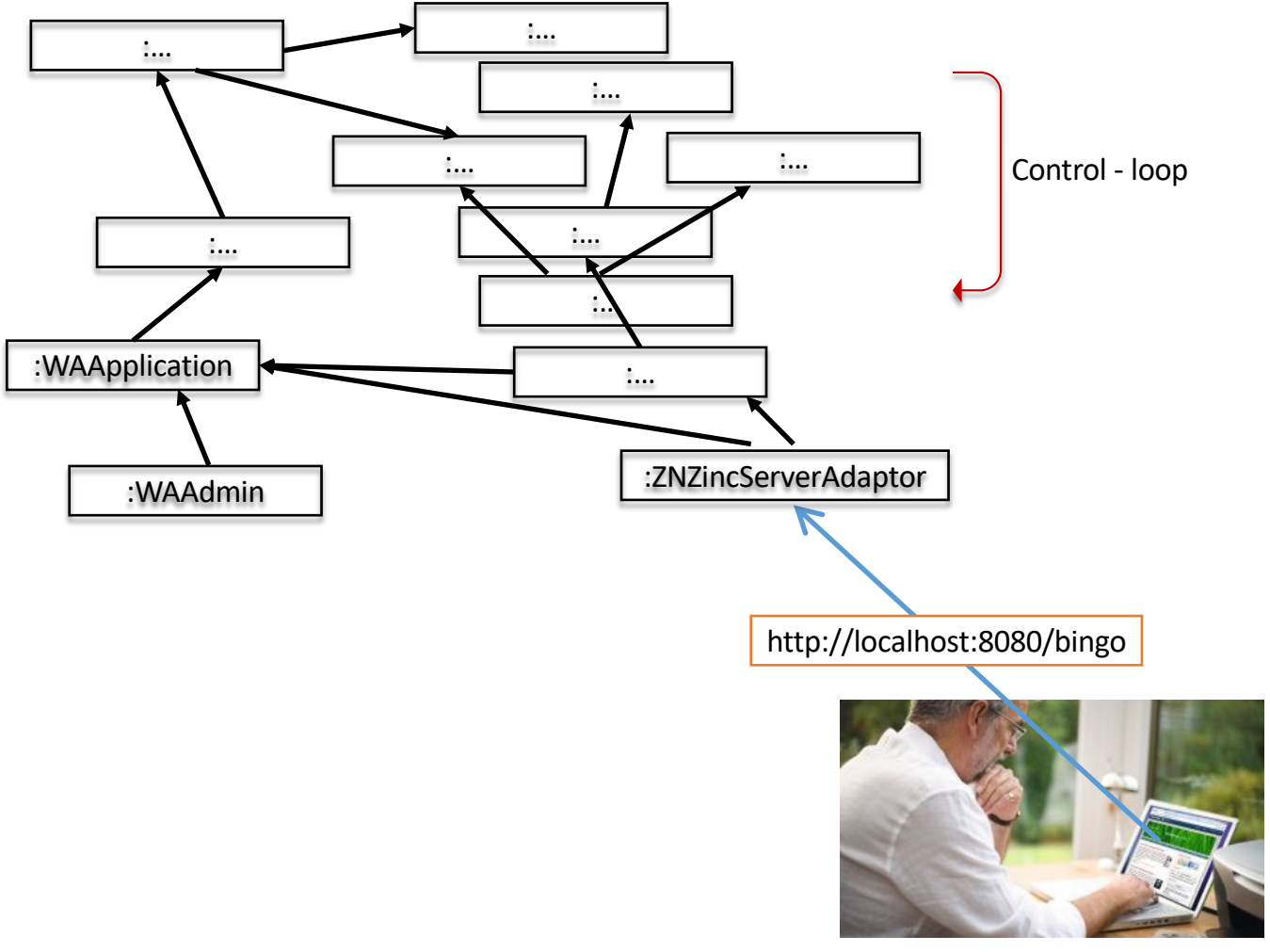
:ZNZincServerAdaptor

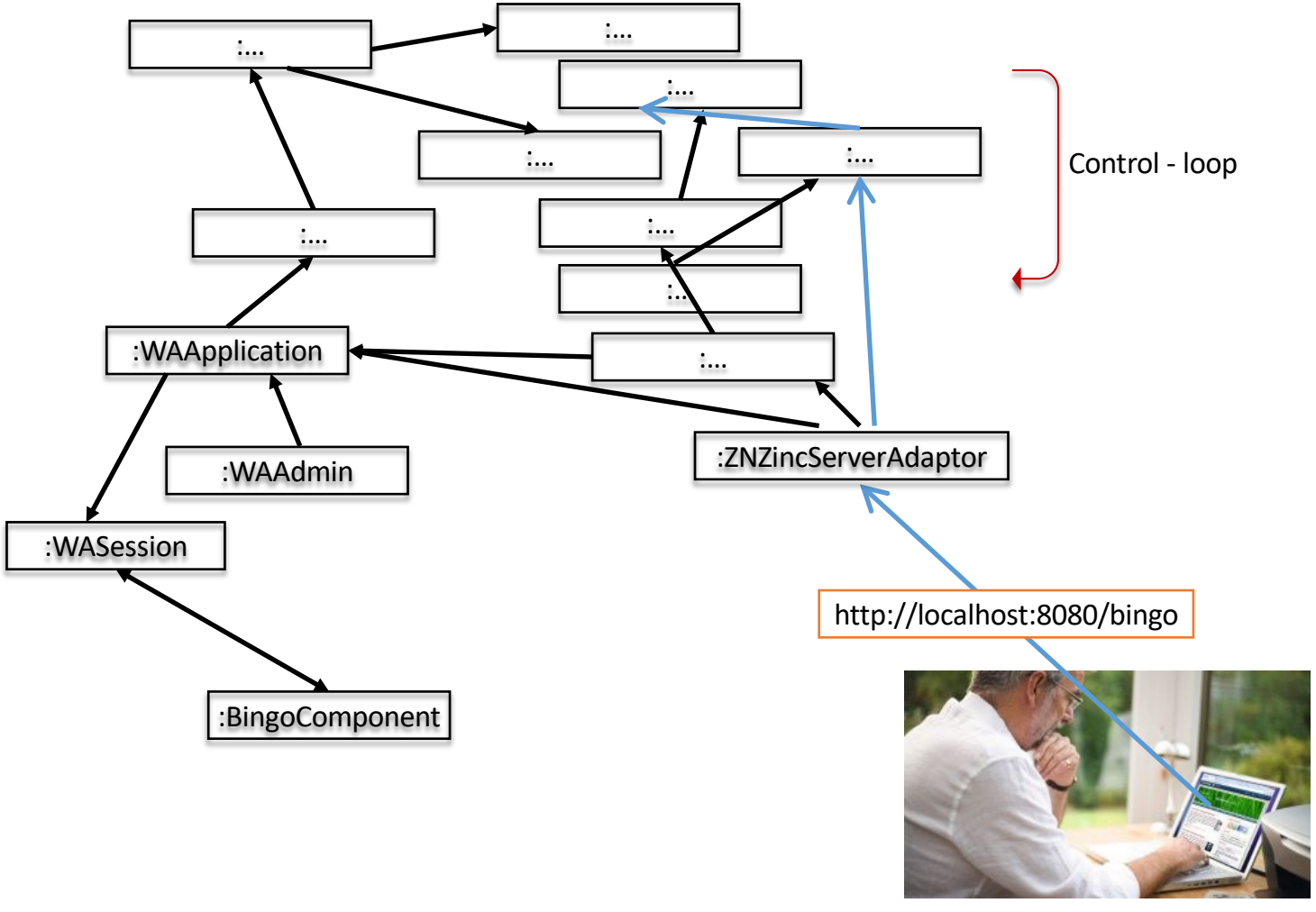

:WAAdmin
:ZNZincServerAdaptor
register(BingoComponent, “bingo”)

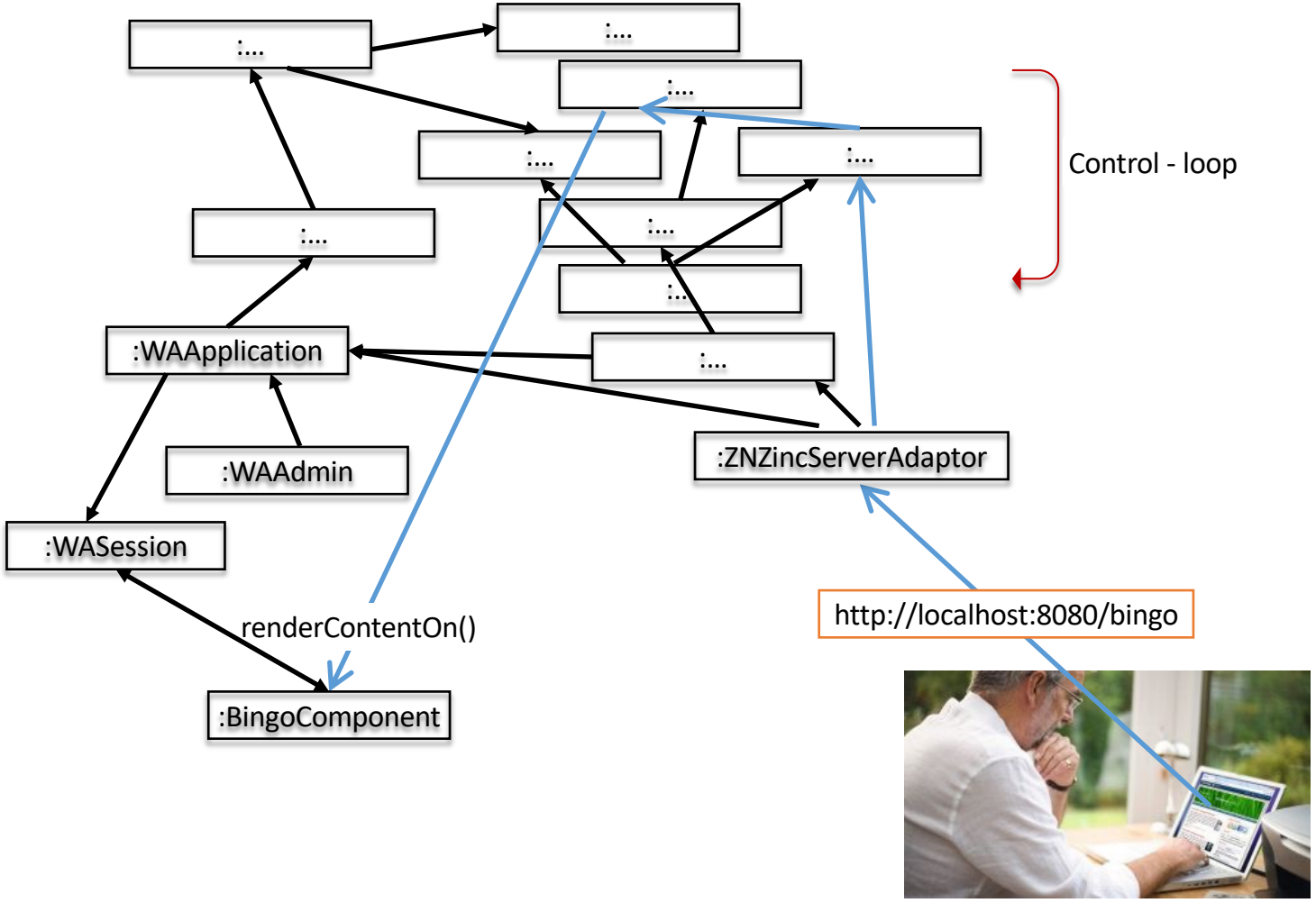

:...
:...
:...
:...
:...
:...
:WAApplication
:WAAdmin
:ZNZincServerAdaptor
register(BingoComponent, “bingo”)

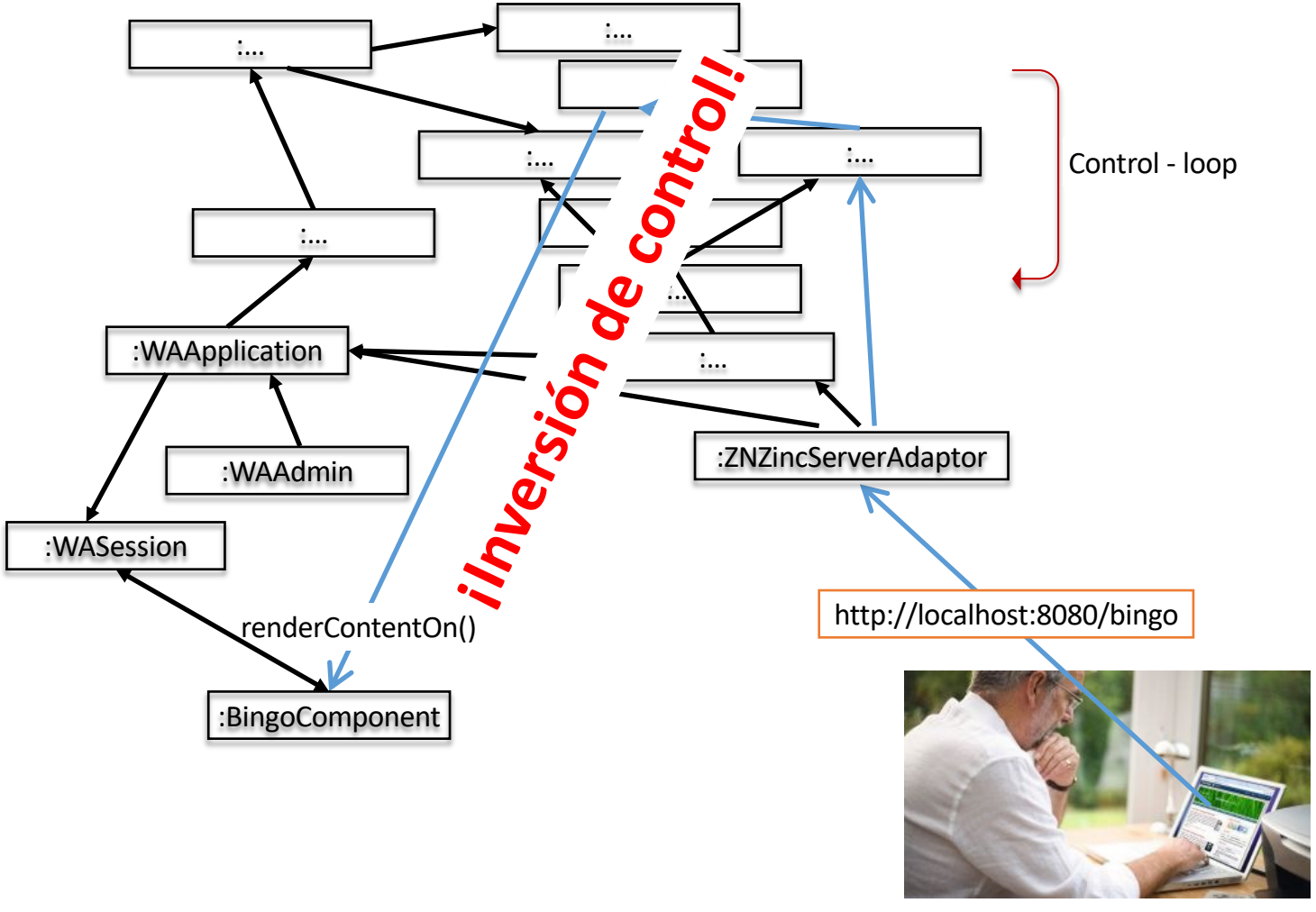

:...
:...
:...
:...
:...
:...
:WAApplication
:WAAdmin
:ZNZincServerAdaptor
start()
Seaside Control Panel
ZnZincServerAdaptor zinc on port 8080 [stopped]
Filter...
Start
Stop
Browse
Type: ZnZincServerAdaptor
Port: 8080


:...
:...
:...
:...
:...
:...
:...
:...
:WAApplication
:...
:WAAdmin
:ZNZincServerAdaptor
Control - loop
start()

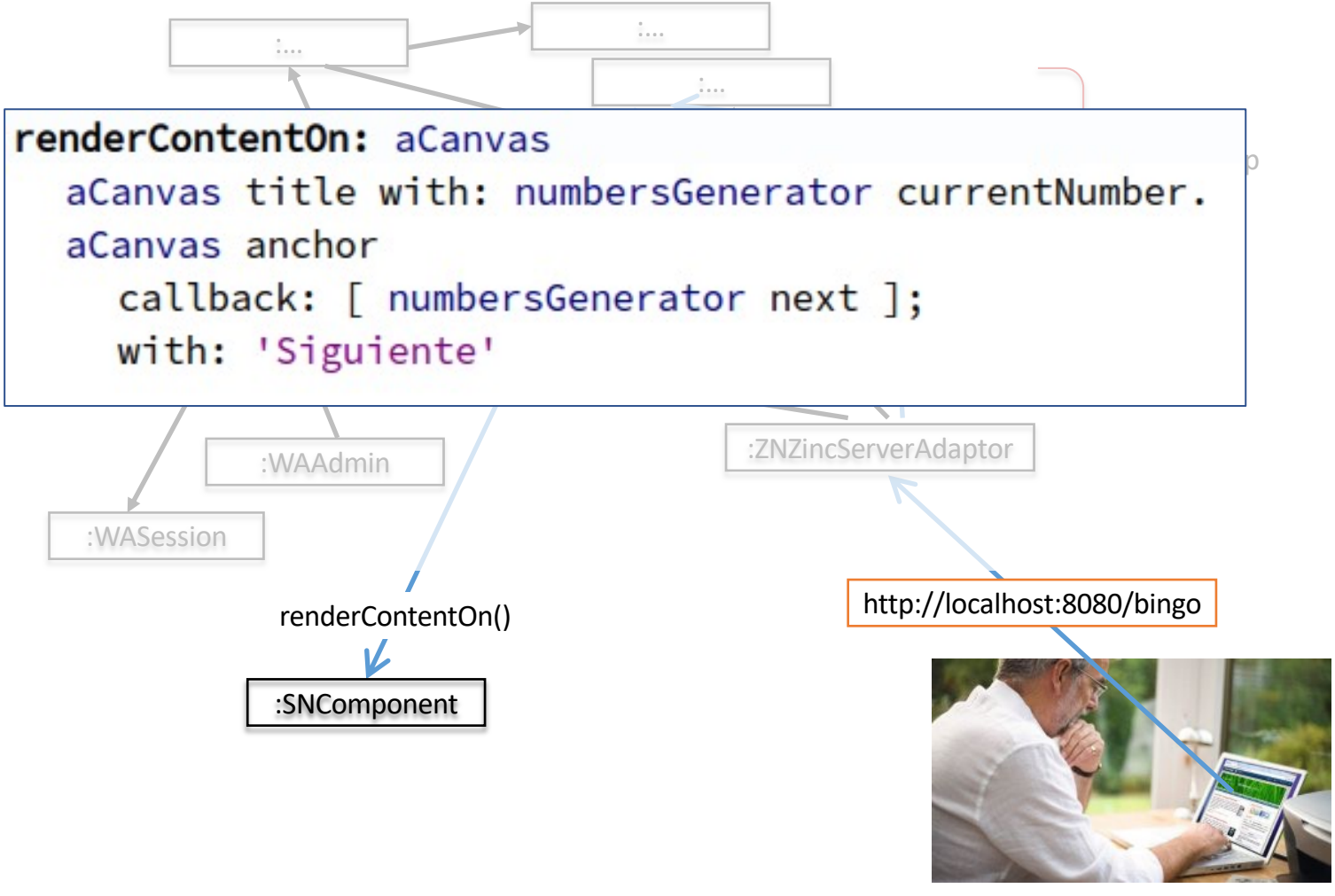

:...
:...
:...
:...
:...
:...
:...
:...
:WAApplication
:...
:WAAdmin
:ZNZincServerAdaptor
Control - loop
http://localhost:8080/bingo


:...
:...
:...
:...
:...
:...
:...
:...
:...
:WAApplication
:WAAdmin
:WASession
:BingoComponent
:ZNZincServerAdaptor
Control - loop
http://localhost:8080/bingo


:...
:...
:...
:...
:...
:...
:...
:...
:...
:WAApplication
:WAAdmin
:WASession
:ZNZincServerAdaptor
renderContentOn()
:BingoComponent
Control - loop
http://localhost:8080/bingo


:...
:...
:...
:...
:...
:...
:...
:WAApplication
:WAAdmin
:WASession
:ZNZincServerAdaptor
¡Inversión de control!
Control - loop
renderContentOn()
:BingoComponent
http://localhost:8080/bingo

Y los callbacks? Donde están los callbacks?


:....
:....
:....
renderContentOn: aCanvas
aCanvas title with: numbersGenerator currentNumber.
aCanvas anchor
callback: [numbersGenerator next];
with: 'Siguiente'
:WAAdmin
:ZNZincServerAdaptor
:WASession
renderContentOn()
http://localhost:8080/bingo
:SNComponent

```
renderContentOn: aCanvas
   aCanvas title with: numbersGenerator currentNumber.
   aCanvas anchor
      callback: [ numbersGenerator next ];
      with: 'Siguiente'
```

:....

:....

:....

:WAAdmin

:ZNZincServerAdaptor

:WASession

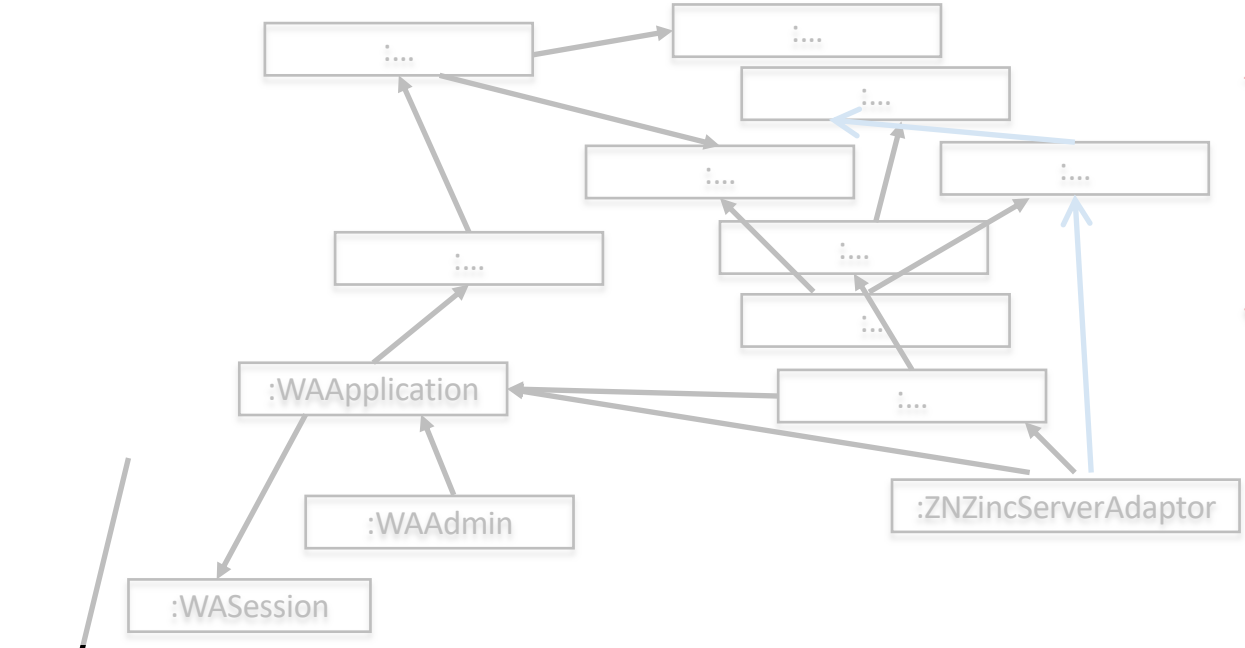
renderContentOn()

:SNComponent

:BlockClosure

```
[ numbersGenerator next ]
```

http://localhost:8080/bingo?_s=pvu-z1FXWEsKe

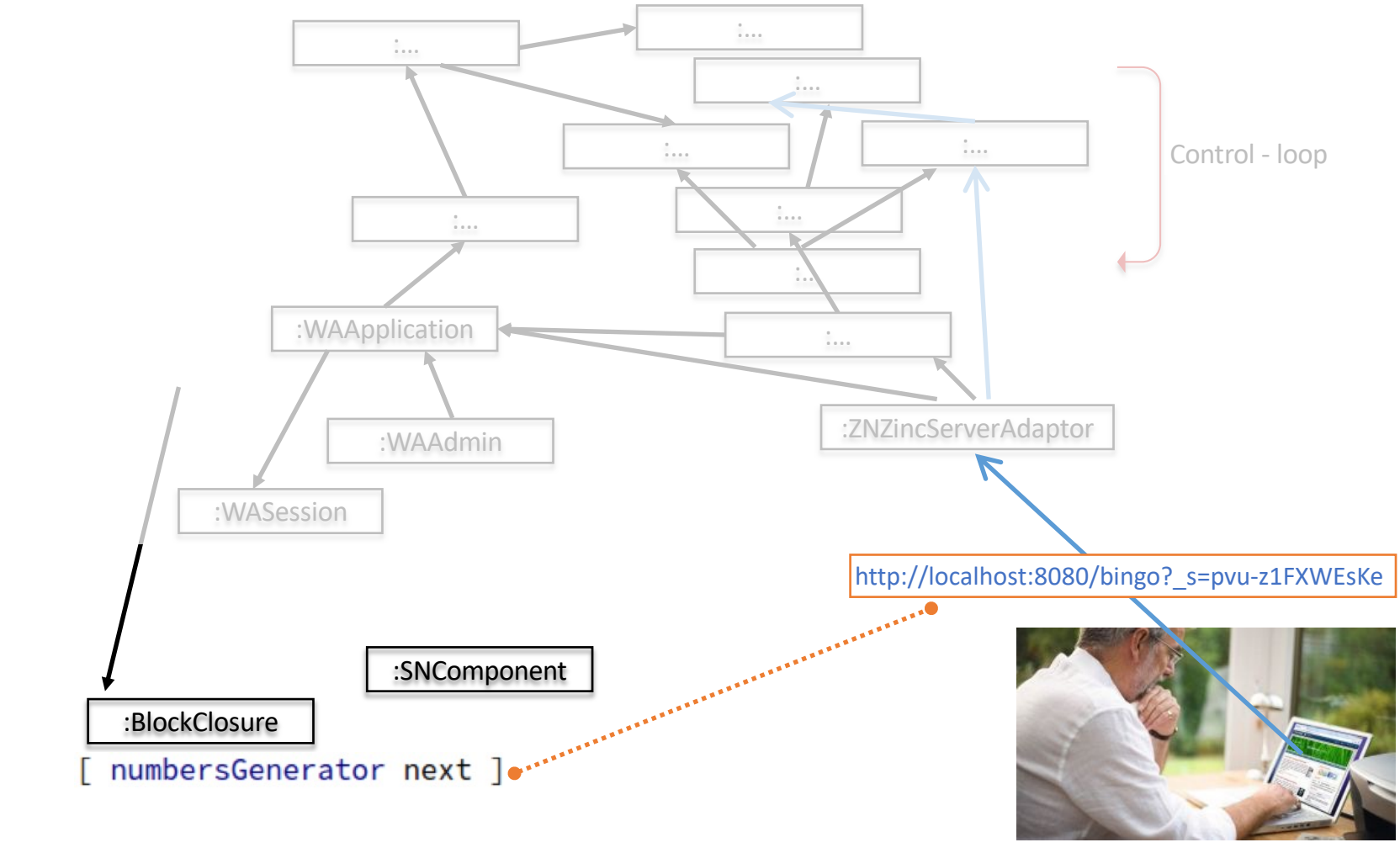


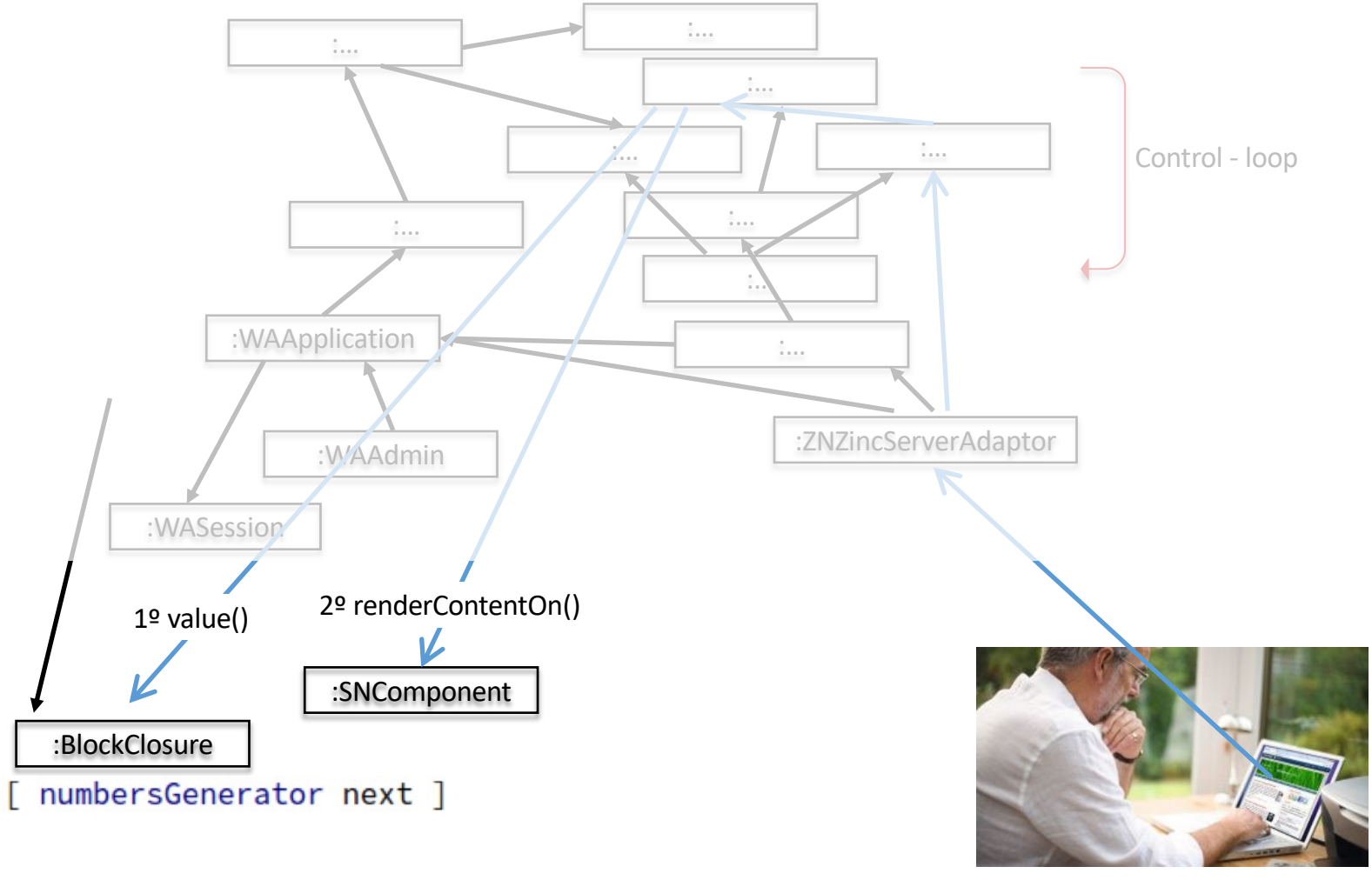
:SNComponent

:BlockClosure

```
[ numbersGenerator next ]
```

http://localhost:8080/bingo?_s=pvu-z1FXWEsKe


:....
:....
:....
:....
:....
:....
:....
:....
Control - loop
:WAApplication
:....
:WAAdmin
:ZNZincServerAdaptor
:WASession
http://localhost:8080/bingo?_s=pvu-z1FXWEsKe
:SNComponent
:BlockClosure
[numbersGenerator next]


:....
:....
:....
:....
:....
:....
:....
:....
:WAApplication
:WAAdmin
:WASession
:ZNZincServerAdaptor
Control - loop
1º value()
2º renderContentOn()
:SNComponent
:BlockClosure
[numbersGenerator next]