RIFE

Modern web application development in Java with Web Continuations

Geert Bevin
CEO
Uwyn

What is RIFE?

- full stack end-to-end framework
- rapid web application development
- implicit standardized application structure
- maintainable and inspectable code-base
- respects HTTP limitations
- cutting edge development features

What does RIFE provide?

- componentized **web application engine**,
- integrated **web continuations**,
- multi-format **template engine** (xhtml, xml, sql, java, text, …),
- template **content transforming** (xslt, filtering, …),
- automated **form building**,
- **validation** framework,
- **authentication** framework,
- database **query builders**,
- fault-tolerant **JDBC wrappers**,
- integrated **connection pooling**,
- web-oriented **database utilities**,
- service-oriented **life-cycle management**,
- cron-like **scheduler**,
- **configuration** framework.

Today's focus

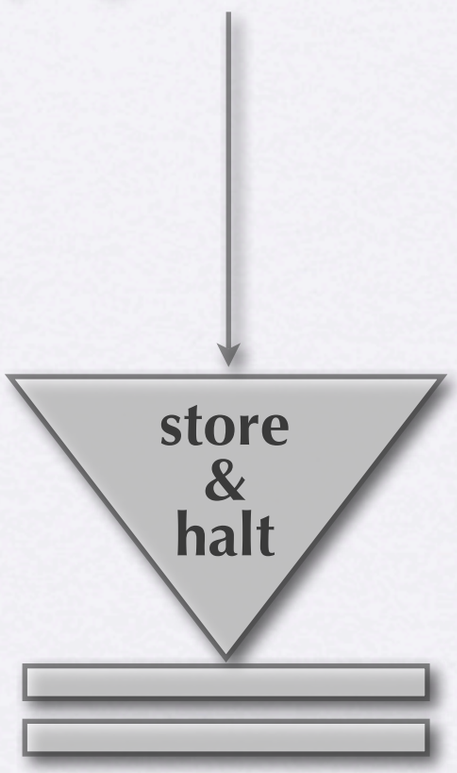
Web continuations

What are web continuations? (1/4)

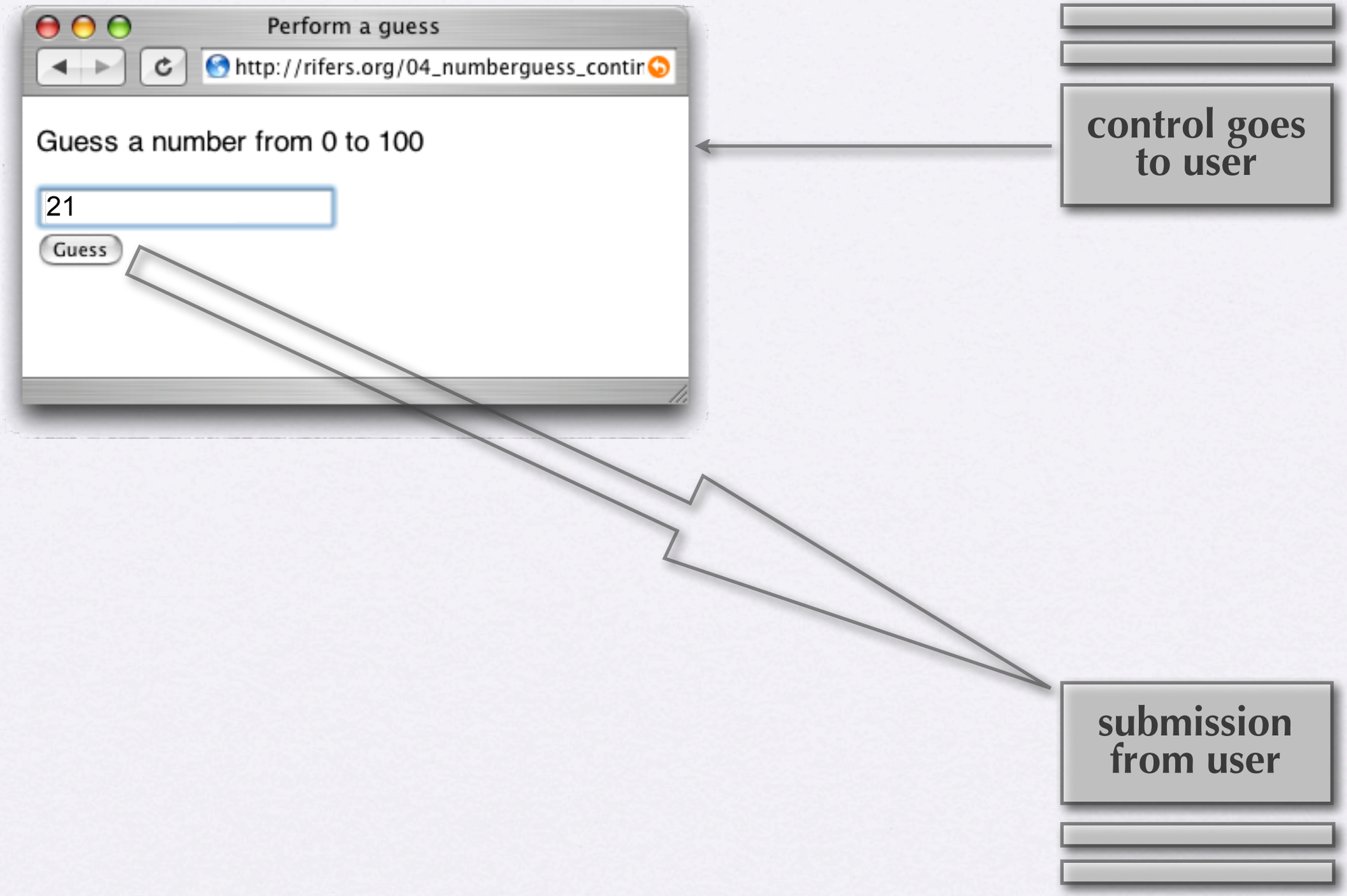
```
public class NumberGuess extends Element
{
    private static Random sRandomNumbers = new Random();

    public void processElement()
    {
        Template template = getHtmlTemplate("guess");

        int answer = 0;
        int guesses = 0;
        int guess = -1;

        synchronized (this) { answer = sRandomNumbers.nextInt(101); }

        while (guess != answer)
        {
            print(template);
            pause();
            guess = getParameterInt("guess", -1);
            if (guess < 0 || guess > 100)
            {
                template.setBlock("warning", "invalid");
                continue;
            }

            guesses++;

            if (answer < guess)      template.setBlock("indication", "lower");
            else if (answer > guess) template.setBlock("indication", "higher");
        }

        template = getHtmlTemplate("success");
        template.setValue("answer", answer);
        template.setValue("guesses", guesses);
        print(template);
    }
}
```

local variable context

program location

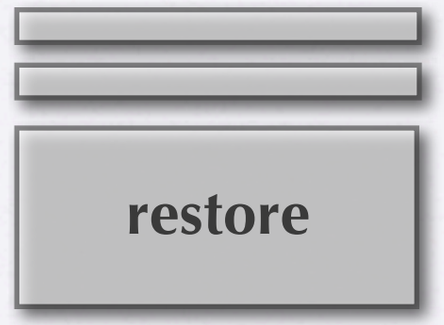


What are web continuations? (2/4)



What are web continuations? (3/4)

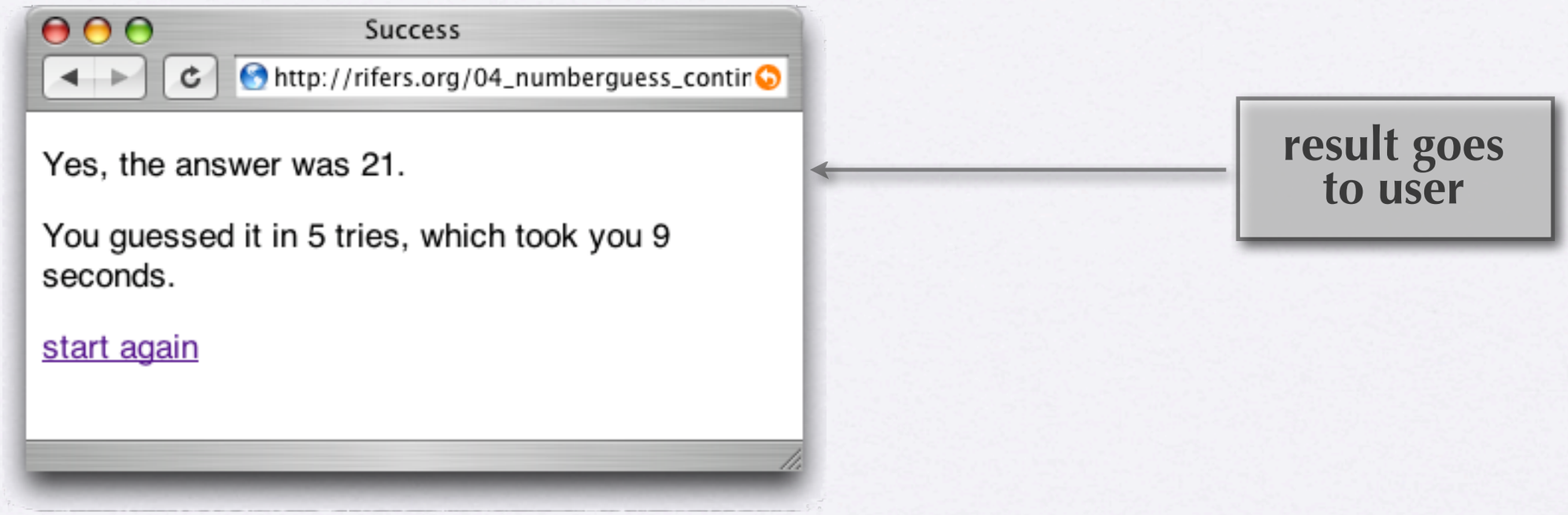
```
public class NumberGuess extends Element
{
    private static Random sRandomNumbers = new Random();

    public void processElement()
    {
        Template template = getHtmlTemplate("guess");

        int answer = 0;
        int guesses = 0;
        int guess = -1;

        synchronized (this) { answer = sRandomNumbers.nextInt(101); }

        while (guess != answer)
        {
            print(template);

            pause();

            guess = getParameterInt("guess", -1);
            if (guess < 0 || guess > 100)
            {
                template.setBlock("warning", "invalid");
                continue;
            }

            guesses++;

            if (answer < guess)      template.setBlock("indication", "lower");
            else if (answer > guess) template.setBlock("indication", "higher");
        }

        template = getHtmlTemplate("success");
        template.setValue("answer", answer);
        template.setValue("guesses", guesses);
        print(template);
    }
}
```



local variable context

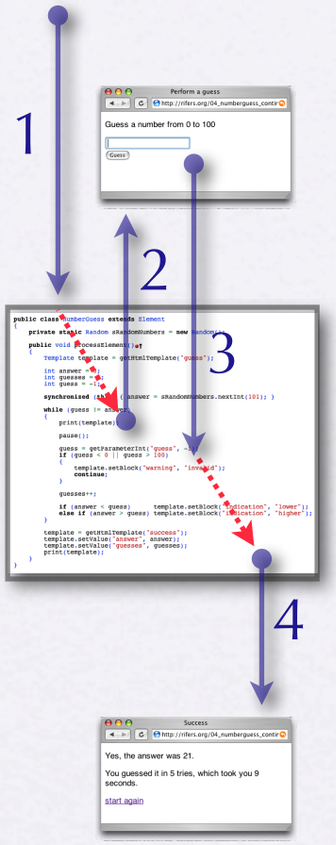
program location

What are web continuations? (4/4)



How does it compare?

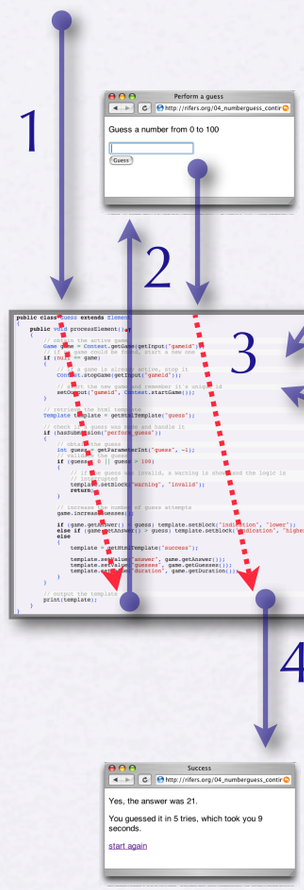
Continuations



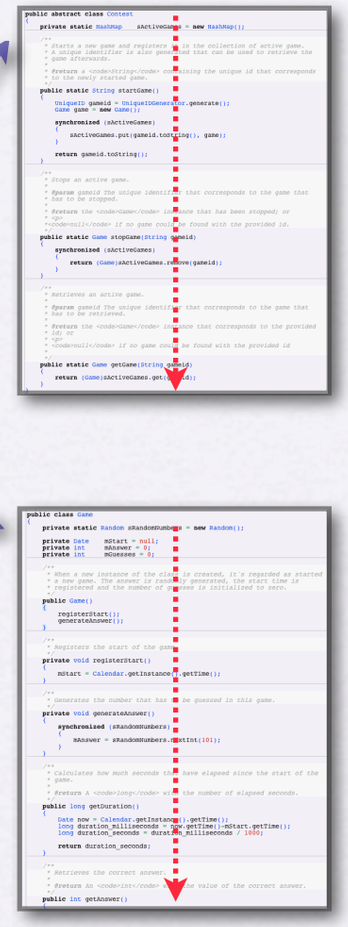
1. process request
2. display form
3. process reply
4. display result

executed 1x
internal state

Traditionally



executed 2x



external state

Benefits of web continuations

- natural control flow
- centralized and automatic state handling
- facilitates creation of complex web application flows

Downsides to web continuations

- cloning of variable context can be expensive
- state is handled by the server-side
- tiny restrictions for the developer

More information

http://rifers.org

http://www.uwyn.com