



## 2. Übung, 2. Aufgabe

```
public class Task {
    public Task(long key, String shortTitle, String description, Date
        startDate, Date endDate) {
        this.key = key; [...] Initialisierung
    } [...]
    public void setStartDate(String startDate) throws ParseException {
        this.startDate = df.parse(startDate); Konvertierung
    }
    public String getStartDateAsString() {
        if (startDate!=null)
            return df.format(startDate) else return new String();
    }
    public String toString() {
        return shortTitle + ": " + description + " (" + startDate + "-> "
            + endDate + ")"; Daten
    }
    private long key = -1; private Date endDate = null; [...]
    private static DateFormat df =
        DateFormat.getDateInstance(DateFormat.SHORT);
}
alternativ:
new SimpleDateFormat("dd.mm.YYYY")
```

Softwarepraktiku

**Parser für  
Datumswerte**



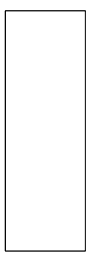
## 2. Übung, 3. Aufgabe

---

```
public class Tasks {
    public Tasks() {
        this.tasks = new ArrayList();
        key=1;
    }
    public Iterator getIterator() {
        return tasks.iterator();
    }
    public void addTask(Task task) {
        task.setKey(key);
        key=key+1;
        tasks.add(task);
    }
    private List tasks = null;
    private long key;
}
```

Liste erzeugen

automatische Schlüsselvergabe





## 2. Übung, 4. Aufgabe

---

```
public class Taskstest {
    public static void main(String[] args) {
        DateFormat df = DateFormat.getDateInstance();
        try {
            Task task1 = new Task(1, "Task1", "Beschreibung für Aufgabe
                1", df.parse("01.10.2002"), df.parse("01.11.2002"));
            Task task2 = new Task(2, "Task2", "Beschreibung für Aufgabe
                2", df.parse("02.10.2002"), df.parse("02.11.2002"));

            Tasks tasks = new Tasks();
            tasks.addTask(task1);
            tasks.addTask(task2);

            for (Iterator iter = tasks.getIterator(); iter.hasNext();
                /* NO-OP */ ) {
                System.out.println((Task) iter.next());
            }
        } catch (ParseException e) {
        }
    }
}
```





## 2. Übung, 5. Aufgabe b)

```
public class ShowTasksPrepare extends HttpServlet {  
    public void init() {  
        ServletContext ctx = getServletContext();  
        Tasks tasks = (Tasks) ctx.getAttribute("tasks");  
        if (tasks == null) {  
            tasks = new Tasks();  
            ctx.setAttribute("tasks", tasks);  
        }  
    }  
    protected void doGet(HttpServletRequest request,  
        HttpServletResponse response) throws ServletException, IOException {  
        ServletContext ctx = getServletContext();  
        Iterator iter = ((Tasks) ctx.getAttribute("tasks")).getIterator();  
        request.setAttribute("task-iterator", iter);  
        RequestDispatcher disp =  
            ctx.getRequestDispatcher("/ShowTasksView.jsp");  
        disp.forward(request, response);  
    }  
}
```

**Datenhaltung im  
Context**

**Iterator mit  
allen Tasks**





## 2. Übung, 5. Aufgabe c)

```
<HTML>
<%@ page import="Task, java.util.Iterator" %>
<BODY BGCOLOR="white">
<TABLE border=0 cellpadding=5>
  <TR bgcolor="DDDDDD">
    <TH>Key</TH><TH>Bezeichnung</TH><TH>Start</TH><TH>Ende</TH>
  </TR>
  <%
    Task task = null;
    for (Iterator iter = (Iterator) request.getAttribute("task-iterator");
        iter.hasNext(); /* NO-OP */ ) {
      task = (Task) iter.next(); %>
    <TR><TD><%= task.getKey()%></TD>
      <TD><%= task.getShortTitle()%></TD>
      <TD><%= task.getStartDateAsString()%></TD>
      <TD><%= task.getEndDateAsString()%></TD>
    </TR>
  <%
    } /* Ends for-loop */
%></TABLE></BODY></HTML>
```

Iterator  
durchlaufen





## 2. Übung, 6. Aufgabe b)

---

```
public class NewTaskExecute extends HttpServlet {
    public void doPost(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException {
        ServletContext ctx = getServletContext();
        Task task = new Task();
        try {
            task.setShortTitle(request.getParameter("short_title"));
            task.setDescription(request.getParameter("description"));
            task.setEndDate(request.getParameter("end_date"));
            task.setStartDate(request.getParameter("start_date"));
        } catch (ParseException e) {}
        Tasks tasks = (Tasks) ctx.getAttribute("tasks");
        tasks.addTask(task);
        response.sendRedirect("showTasksPrepare");
    }
}
```

**Aufgabe anlegen**

**und  
abspeichern**

