TEX and PDF in Education

M. Ćirić and G. Kljajić

Department of Mathematics & Informatics

Faculty of Sciences & Mathematics

University of Niš, Serbia

A Short Introduction to TEX

TEX is a computer language, created by Donald Knuth, intended to produce high-quality typesetting.

TEX sets normal text beautifully, with linebreaking algorithms that are noticeably better than those used by common word processors.

But TEX really excels when it comes to setting extremely complicated material such as is common in science, engineering and mathematics.

Unlike the documents created by popular word processors which often become unusable when the program is upgraded, documents formatted using TEX are extremely stable, because TEX files are plain text.

A Short Introduction to TEX

TEX is a computer language, created by Donald Knuth, intended to produce high-quality typesetting.

TEX sets normal text beautifully, with linebreaking algorithms that are noticeably better than those used by common word processors.

But TEX really excels when it comes to setting extremely complicated material such as is common in science, engineering and mathematics.

Unlike the documents created by popular word processors which often become unusable when the program is upgraded, documents formatted using TEX are extremely stable, because TEX files are plain text.

A Short Introduction to TEX

TEX is a computer language, created by Donald Knuth, intended to produce high-quality typesetting.

TEX sets normal text beautifully, with linebreaking algorithms that are noticeably better than those used by common word processors.

But TEX really excels when it comes to setting extremely complicated material such as is common in science, engineering and mathematics.

Unlike the documents created by popular word processors which often become unusable when the program is upgraded, documents formatted using TEX are extremely stable, because TEX files are plain text.

A Short Introduction to TEX

TEX is a computer language, created by Donald Knuth, intended to produce high-quality typesetting.

TEX sets normal text beautifully, with linebreaking algorithms that are noticeably better than those used by common word processors.

But TEX really excels when it comes to setting extremely complicated material such as is common in science, engineering and mathematics.

Unlike the documents created by popular word processors which often become unusable when the program is upgraded, documents formatted using TEX are extremely stable, because TEX files are plain text.

The TEX program itself is a macro compiler, and it's input consists of a stream of mixed macro commands and text.

The TEX system has precise knowledge of the sizes of all characters and symbols, and using this information, it computes the optimal arrangement of letters per line and lines per page.

It then produces a DVI file (for "device independent"), as it's primary output format, containing the final locations of all characters.

This DVI file can be printed directly given an appropriate printer driver, viewed using some of many existing viewers, or it can be converted to other formats.

It can be converted to a printer language such as PostScript, or to a web language such as PDF or HTML, or, probably, to whatever will appear in the future. And, the typesetting (line breaks, etc.) will be the same no matter where our output appears.

The TEX program itself is a macro compiler, and it's input consists of a stream of mixed macro commands and text.

The TEX system has precise knowledge of the sizes of all characters and symbols, and using this information, it computes the optimal arrangement of letters per line and lines per page.

It then produces a DVI file (for "device independent"), as it's primary output format, containing the final locations of all characters.

This DVI file can be printed directly given an appropriate printer driver, viewed using some of many existing viewers, or it can be converted to other formats.

It can be converted to a printer language such as PostScript, or to a web language such as PDF or HTML, or, probably, to whatever will appear in the future. And, the typesetting (line breaks, etc.) will be the same no matter where our output appears.

The TEX program itself is a macro compiler, and it's input consists of a stream of mixed macro commands and text.

The TEX system has precise knowledge of the sizes of all characters and symbols, and using this information, it computes the optimal arrangement of letters per line and lines per page.

It then produces a DVI file (for "device independent"), as it's primary output format, containing the final locations of all characters.

This DVI file can be printed directly given an appropriate printer driver, viewed using some of many existing viewers, or it can be converted to other formats.

It can be converted to a printer language such as PostScript, or to a web language such as PDF or HTML, or, probably, to whatever will appear in the future. And, the typesetting (line breaks, etc.) will be the same no matter where our output appears.

The TEX program itself is a macro compiler, and it's input consists of a stream of mixed macro commands and text.

The TEX system has precise knowledge of the sizes of all characters and symbols, and using this information, it computes the optimal arrangement of letters per line and lines per page.

It then produces a DVI file (for "device independent"), as it's primary output format, containing the final locations of all characters.

This DVI file can be printed directly given an appropriate printer driver, viewed using some of many existing viewers, or it can be converted to other formats.

It can be converted to a printer language such as PostScript, or to a web language such as PDF or HTML, or, probably, to whatever will appear in the future. And, the typesetting (line breaks, etc.) will be the same no matter where our output appears.

The TEX program itself is a macro compiler, and it's input consists of a stream of mixed macro commands and text.

The TEX system has precise knowledge of the sizes of all characters and symbols, and using this information, it computes the optimal arrangement of letters per line and lines per page.

It then produces a DVI file (for "device independent"), as it's primary output format, containing the final locations of all characters.

This DVI file can be printed directly given an appropriate printer driver, viewed using some of many existing viewers, or it can be converted to other formats.

It can be converted to a printer language such as PostScript, or to a web language such as PDF or HTML, or, probably, to whatever will appear in the future. And, the typesetting (line breaks, etc.) will be the same no matter where our output appears.

Numerous extensions to TEX exist:

LATEX is an extension of TEX whose major features include a strong focus on document structure and the logical markup of text, automatic numbering and cross-referencing, and much more.

It was originally written in 1984 by Leslie Lamport and has become the dominant method for using TEX.

The current version is LATEX2ε developed by the LATEX3 team – Frank Mittelbach (project leader), David Carlisle and others.

AMSTEX (produced by the American Mathematical Society) provides many features to make typesetting mathematics convenient while meeting the standards of the AMS for publication.

AMSTEX provides many additional mathematical constructs and fonts with many more mathematical symbols than the fonts that come with TEX.

Numerous extensions to TEX exist:

- **LATEX is an extension of TEX whose major features include a strong focus on document structure and the logical markup of text, automatic numbering and cross-referencing, and much more.**

  It was originally written in 1984 by Leslie Lamport and has become the dominant method for using TEX.

  The current version is LATEX2ε developed by the LATEX3 team – Frank Mittelbach (project leader), David Carlisle and others.

- **AMSTEX (produced by the American Mathematical Society) provides many features to make typesetting mathematics convenient while meeting the standards of the AMS for publication.**

  AMSTEX provides many additional mathematical constructs and fonts with many more mathematical symbols than the fonts that come with TEX.

Numerous extensions to TEX exist:

LATEX is an extension of TEX whose major features include a strong focus on document structure and the logical markup of text, automatic numbering and cross-referencing, and much more.

It was originally written in 1984 by Leslie Lamport and has become the dominant method for using TEX.

The current version is LATEX2ε developed by the LATEX3 team – Frank Mittelbach (project leader), David Carlisle and others.

AMSTEX (produced by the American Mathematical Society) provides many features to make typesetting mathematics convenient while meeting the standards of the AMS for publication.

AMSTEX provides many additional mathematical constructs and fonts with many more mathematical symbols than the fonts that come with TEX.

Numerous extensions to TEX exist:

LATEX is an extension of TEX whose major features include a strong focus on document structure and the logical markup of text, automatic numbering and cross-referencing, and much more.

It was originally written in 1984 by Leslie Lamport and has become the dominant method for using TEX.

The current version is LATEX2ε developed by the LATEX3 team – Frank Mittelbach (project leader), David Carlisle and others.

AMSTEX (produced by the American Mathematical Society) provides many features to make typesetting mathematics convenient while meeting the standards of the AMS for publication.

AMSTEX provides many additional mathematical constructs and fonts with many more mathematical symbols than the fonts that come with TEX.

Numerous extensions to TEX exist:

LATEX is an extension of TEX whose major features include a strong focus on document structure and the logical markup of text, automatic numbering and cross-referencing, and much more.

It was originally written in 1984 by Leslie Lamport and has become the dominant method for using TEX.

The current version is LATEX2ε developed by the LATEX3 team – Frank Mittelbach (project leader), David Carlisle and others.

AMSTEX (produced by the American Mathematical Society) provides many features to make typesetting mathematics convenient while meeting the standards of the AMS for publication.

AMSTEX provides many additional mathematical constructs and fonts with many more mathematical symbols than the fonts that come with TEX.

Numerous extensions to TEX exist:

LATEX is an extension of TEX whose major features include a strong focus on document structure and the logical markup of text, automatic numbering and cross-referencing, and much more.

It was originally written in 1984 by Leslie Lamport and has become the dominant method for using TEX.

The current version is LATEX2ε developed by the LATEX3 team – Frank Mittelbach (project leader), David Carlisle and others.

AMSTEX (produced by the American Mathematical Society) provides many features to make typesetting mathematics convenient while meeting the standards of the AMS for publication.

AMSTEX provides many additional mathematical constructs and fonts with many more mathematical symbols than the fonts that come with TEX.

AMSLaTeX is a common extension of LaTeX and AMSTeX obtained by combining the features of AMSTeX with the ones of LaTeX.

AMSLaTeX provides all of the functionality of LaTeX (as its extension) and the functionality of AMSTeX in LaTeX syntax and access to additional mathematical constructs and mathematical symbols not present in LaTeX.

In LaTeX 2ε this is achieved using amsmath, amsfonts and amscls document class packages.

AMS-LaTeX is a common extension of LaTeX and AMS-TeX obtained by combining the features of AMS-TeX with the ones of LaTeX.

AMS-LaTeX provides all of the functionality of LaTeX (as its extension) and the functionality of AMS-TeX in LaTeX syntax and access to additional mathematical constructs and mathematical symbols not present in LaTeX.

In LaTeX 2ε this is achieved using amsmath, amsfonts and amscls document class packages.

AMS-LaTeX is a common extension of LaTeX and AMSTeX obtained by combining the features of AMSTeX with the ones of LaTeX.

AMS-LaTeX provides all of the functionality of LaTeX (as its extension) and the functionality of AMSTeX in LaTeX syntax and access to additional mathematical constructs and mathematical symbols not present in LaTeX.

In LaTeX 2ε this is achieved using amsmath, amsfonts and amscls document class packages.

TEX and PDF

Although TEX has traditionally generated DVI as its primary output format, TEX's creator Donald Knuth has himself said that an alternative output format has to be used. He had PostScript in mind, but PDF turned out to be an even better candidate for such purposes.

It is a very compact language, already well-established as a de facto standard portable document format both on and off the web.

Its deliberate omission of the procedural elements of PostScript ensures that it is efficient enough to be used for direct screen display as well as for less time-critical applications such as printing.

Besides, PDF has many other advantages – numerous dynamic effects, interactivity etc.

TEX and PDF

Although TEX has traditionally generated DVI as its primary output format, TEX's creator Donald Knuth has himself said that an alternative output format has to be used. He had PostScript in mind, but PDF turned out to be an even better candidate for such purposes.

It is a very compact language, already well-established as a de facto standard portable document format both on and off the web.

Its deliberate omission of the procedural elements of PostScript ensures that it is efficient enough to be used for direct screen display as well as for less time-critical applications such as printing.

Besides, PDF has many other advantages – numerous dynamic effects, interactivity etc.

TEX and PDF

Although TEX has traditionally generated DVI as its primary output format, TEX's creator Donald Knuth has himself said that an alternative output format has to be used. He had PostScript in mind, but PDF turned out to be an even better candidate for such purposes.

It is a very compact language, already well-established as a de facto standard portable document format both on and off the web.

Its deliberate omission of the procedural elements of PostScript ensures that it is efficient enough to be used for direct screen display as well as for less time-critical applications such as printing.

Besides, PDF has many other advantages – numerous dynamic effects, interactivity etc.

TEX and PDF

Although TEX has traditionally generated DVI as its primary output format, TEX's creator Donald Knuth has himself said that an alternative output format has to be used. He had PostScript in mind, but PDF turned out to be an even better candidate for such purposes.

It is a very compact language, already well-established as a de facto standard portable document format both on and off the web.

Its deliberate omission of the procedural elements of PostScript ensures that it is efficient enough to be used for direct screen display as well as for less time-critical applications such as printing.

Besides, PDF has many other advantages – numerous dynamic effects, interactivity etc.

The oldest way for creating PDF from TeX or LaTeX source consists of the three steps:

(1) **compiling TeX source file to DVI file which contains `\special` commands for PDF support;**

(2) **converting DVI file to PostScript by some DVI-to-PostScript driver, such as dvips or dvipsone;**

(3) **translating PostScript file to PDF by some PostScript-to-PDF translator, such as Acrobat Distiller or Ghostscript.**

Another way is to use dvipdf or dvipdfm, DVI-to-PDF drivers made by Sergey Lesenko, which can simplify this process by eliminating the need for PostScript generation.

Probably the best way is to use pdfTeX or pdfLaTeX created by Han The Thanh, which can produce PDF output from a TeX or LaTeX source without generating DVI.

The oldest way for creating PDF from TeX or LaTeX source consists of the three steps:

(1) **compiling TeX source file to DVI file which contains `\special` commands for PDF support;**

(2) converting DVI file to PostScript by some DVI-to-PostScript driver, such as **dvips** or **dvipsone**;

(3) translating PostScript file to PDF by some PostScript-to-PDF translator, such as **Acrobat Distiller** or **Ghostscript**.

Another way is to use **dvipdf** or **dvipdfm**, DVI-to-PDF drivers made by Sergey Lesenko, which can simplify this process by eliminating the need for PostScript generation.

Probably the best way is to use **pdfTeX** or **pdfLaTeX** created by Han The Thanh, which can produce PDF output from a TeX or LaTeX source without generating DVI.

The oldest way for creating PDF from TeX or LaTeX source consists of the three steps:

(1) compiling TeX source file to DVI file which contains `\special` commands for PDF support;

(2) converting DVI file to PostScript by some DVI-to-PostScript driver, such as dvips or dvipsone;

(3) translating PostScript file to PDF by some PostScript-to-PDF translator, such as Acrobat Distiller or Ghostscript.

Another way is to use dvipdf or dvipdfm, DVI-to-PDF drivers made by Sergey Lesenko, which can simplify this process by eliminating the need for PostScript generation.

Probably the best way is to use pdfTeX or pdfLaTeX created by Han The Thanh, which can produce PDF output from a TeX or LaTeX source without generating DVI.

The oldest way for creating PDF from TeX or LaTeX source consists of the three steps:

(1) compiling TeX source file to DVI file which contains `\special` commands for PDF support;

(2) converting DVI file to PostScript by some DVI-to-PostScript driver, such as **dvips** or **dvipsone**;

(3) translating PostScript file to PDF by some PostScript-to-PDF translator, such as **Acrobat Distiller** or **Ghostscript**.

Another way is to use **dvipdf** or **dvipdfm**, DVI-to-PDF drivers made by Sergey Lesenko, which can simplify this process by eliminating the need for PostScript generation.

Probably the best way is to use **pdfTeX** or **pdfLaTeX** created by Han The Thanh, which can produce PDF output from a TeX or LaTeX source without generating DVI.

The oldest way for creating PDF from TeX or LaTeX source consists of the three steps:

(1) compiling TeX source file to DVI file which contains `\special` commands for PDF support;

(2) converting DVI file to PostScript by some DVI-to-PostScript driver, such as **dvips** or **dvipsone**;

(3) translating PostScript file to PDF by some PostScript-to-PDF translator, such as **Acrobat Distiller** or **Ghostscript**.

Another way is to use dvipdf or dvipdfm, DVI-to-PDF drivers made by Sergey Lesenko, which can simplify this process by eliminating the need for PostScript generation.

Probably the best way is to use **pdfTeX** or **pdfLaTeX** created by Han The Thanh, which can produce PDF output from a TeX or LaTeX source without generating DVI.

The oldest way for creating PDF from TEX or LATEX source consists of the three steps:

(1) compiling TEX source file to DVI file which contains `\special` commands for PDF support;

(2) converting DVI file to PostScript by some DVI-to-PostScript driver, such as **dvips** or **dvipsone**;

(3) translating PostScript file to PDF by some PostScript-to-PDF translator, such as **Acrobat Distiller** or **Ghostscript**.

Another way is to use **dvipdf** or **dvipdfm**, DVI-to-PDF drivers made by Sergey Lesenko, which can simplify this process by eliminating the need for PostScript generation.

Probably the best way is to use pdfTEX or pdfLATEX created by Han The Thanh, which can produce PDF output from a TEX or LATEX source without generating DVI.

PDF in The Classroom

We will present several ways to use PDF in the classroom.

PDF Slide Presentations: They can be made thanks to various dynamic and navigation effects present in the PDF.

We will mention several packages and tools for creating PDF slide presentations.

Exercises and Quizzes: They can be made thanks to the interactivity and the form support in the PDF.

We will mention two systems for creating exercises and quizzes:

- **AcroTEX eDucation Bundle** (University of Akron, USA);
- **MacQTeX** (Macquarie University, Sydney, Australia).

PDF in The Classroom

We will present several ways to use PDF in the classroom.

PDF Slide Presentations: They can be made thanks to various dynamic and navigation effects present in the PDF.

We will mention several packages and tools for creating PDF slide presentations.

Exercises and Quizzes: They can be made thanks to the interactivity and the form support in the PDF.

We will mention two systems for creating exercises and quizzes:

- AcroTEX eDucation Bundle (University of Akron, USA);
- MacQTeX (Macquarie University, Sydney, Australia).

PDF in The Classroom

We will present several ways to use PDF in the classroom.

PDF Slide Presentations: They can be made thanks to various dynamic and navigation effects present in the PDF.

We will mention several packages and tools for creating PDF slide presentations.

Exercises and Quizzes: They can be made thanks to the interactivity and the form support in the PDF.

We will mention two systems for creating exercises and quizzes:

- **AcroTEX eDucation Bundle** (University of Akron, USA);
- **MacQTeX** (Macquarie University, Sydney, Australia).

PDF in The Classroom

We will present several ways to use PDF in the classroom.

PDF Slide Presentations: They can be made thanks to various dynamic and navigation effects present in the PDF.

We will mention several packages and tools for creating PDF slide presentations.

Exercises and Quizzes: They can be made thanks to the interactivity and the form support in the PDF.

We will mention two systems for creating exercises and quizzes:

- AcroTEX eDucation Bundle (University of Akron, USA);
- MacQTeX (Macquarie University, Sydney, Australia).

PDF in The Classroom

We will present several ways to use PDF in the classroom.

PDF Slide Presentations: They can be made thanks to various dynamic and navigation effects present in the PDF.

We will mention several packages and tools for creating PDF slide presentations.

Exercises and Quizzes: They can be made thanks to the interactivity and the form support in the PDF.

We will mention two systems for creating exercises and quizzes:

- **AcroTEX eDucation Bundle** (University of Akron, USA);
- **MacQTeX** (Macquarie University, Sydney, Australia).

PDF in The Classroom

We will present several ways to use PDF in the classroom.

PDF Slide Presentations: They can be made thanks to various dynamic and navigation effects present in the PDF.

We will mention several packages and tools for creating PDF slide presentations.

Exercises and Quizzes: They can be made thanks to the interactivity and the form support in the PDF.

We will mention two systems for creating exercises and quizzes:

- **AcroTEX eDucation Bundle** (University of Akron, USA);
- **MacQTeX** (Macquarie University, Sydney, Australia).

PDF in The Classroom

We will present several ways to use PDF in the classroom.

PDF Slide Presentations: They can be made thanks to various dynamic and navigation effects present in the PDF.

We will mention several packages and tools for creating PDF slide presentations.

Exercises and Quizzes: They can be made thanks to the interactivity and the form support in the PDF.

We will mention two systems for creating exercises and quizzes:

- **AcroTEX eDucation Bundle** (University of Akron, USA);
- **MacQTeX** (Macquarie University, Sydney, Australia).

Presentation Tools

The traditional way to create presentations using LaTeX is to use such LaTeX classes as foiltex or seminar. These were originally designed to create overhead tranparencies.

In the age of laptop presentations, PowerPoint seems to dominate.

But PowerPoint does not handle the complexities of mathematics.

Because of that, various packages were made that can aid the development of presentations with complex mathematical formulas rich in color and special effects.

Presentation Tools

The traditional way to create presentations using LaTeX is to use such LaTeX classes as foiltex or seminar. These were originally designed to create overhead tranparencies.

In the age of laptop presentations, PowerPoint seems to dominate.

But PowerPoint does not handle the complexities of mathematics.

Because of that, various packages were made that can aid the development of presentations with complex mathematical formulas rich in color and special effects.

Presentation Tools

The traditional way to create presentations using LaTeX is to use such LaTeX classes as foiltex or seminar. These were originally designed to create overhead tranparencies.

In the age of laptop presentations, PowerPoint seems to dominate.

But PowerPoint does not handle the complexities of mathematics.

Because of that, various packages were made that can aid the development of presentations with complex mathematical formulas rich in color and special effects.

Presentation Tools

The traditional way to create presentations using LaTeX is to use such LaTeX classes as foiltex or seminar. These were originally designed to create overhead tranparencies.

In the age of laptop presentations, PowerPoint seems to dominate.

But PowerPoint does not handle the complexities of mathematics.

Because of that, various packages were made that can aid the development of presentations with complex mathematical formulas rich in color and special effects.

Generally, there are two kinds of these packages:

Slide Development Packages are LaTeX document classes and other accessories which define PDF specials producing PDF presentations with various dynamic effects – background colors and gradients, transitions effects and step by step presentation of talking points.

Slide Enhancement Tools are either programs which are used to post-process presentations in PostScript or PDF format made by other slide development packages, in order to provide some additional dynamic effects, or are add-ons to other document classes which create dynamic effects.

We will mention the most widely used slide development packages and tools.

Generally, there are two kinds of these packages:

Slide Development Packages are LaTeX document classes and other accessories which define PDF specials producing PDF presentations with various dynamic effects – background colors and gradients, transitions effects and step by step presentation of talking points.

Slide Enhancement Tools are either programs which are used to post-process presentations in PostScript or PDF format made by other slide development packages, in order to provide some additional dynamic effects, or are add-ons to other document classes which create dynamic effects.

We will mention the most widely used slide development packages and tools.

Generally, there are two kinds of these packages:

Slide Development Packages are LaTeX document classes and other accessories which define PDF specials producing PDF presentations with various dynamic effects – background colors and gradients, transitions effects and step by step presentation of talking points.

Slide Enhancement Tools are either programs which are used to post-process presentations in PostScript or PDF format made by other slide development packages, in order to provide some additional dynamic effects, or are add-ons to other document classes which create dynamic effects.

We will mention the most widely used slide development packages and tools.

Generally, there are two kinds of these packages:

Slide Development Packages are LaTeX document classes and other accessories which define PDF specials producing PDF presentations with various dynamic effects – background colors and gradients, transitions effects and step by step presentation of talking points.

Slide Enhancement Tools are either programs which are used to post-process presentations in PostScript or PDF format made by other slide development packages, in order to provide some additional dynamic effects, or are add-ons to other document classes which create dynamic effects.

We will mention the most widely used slide development packages and tools.

Slide Development Packages

FoilTEX and **Seminar** (written by Timothy van Zandt) are LATEX document classes which were originally designed to create overhead transparencies, but with some additional class packages and/or post-processing various dynamic and interactive features can be added.

pdfscreen and **pdfslide** are LATEX document classes implemented by C. V. Radhakrishnan. They are used for online readable documents (pdfscreen) and slide presentations (pdfscreen).

Both of these two packages require the color and hyperref document classes.

Slide Development Packages

FoilTEX and **Seminar** (written by Timothy van Zandt) are LATEX document classes which were originally designed to create overhead transparencies, but with some additional class packages and/or post-processing various dynamic and interactive features can be added.

pdfscreen and **pdfslide** are LATEX document classes implemented by C. V. Radhakrishnan. They are used for online readable documents (pdfscreen) and slide presentations (pdfscreen).

Both of these two packages require the color and hyperref document classes.

Slide Development Packages

FoilTEX and **Seminar** (written by Timothy van Zandt) are LATEX document classes which were originally designed to create overhead transparencies, but with some additional class packages and/or post-processing various dynamic and interactive features can be added.

pdfscreen and **pdfslide** are LATEX document classes implemented by C. V. Radhakrishnan. They are used for online readable documents (pdfscreen) and slide presentations (pdfscreen).

Both of these two packages require the color and hyperref document classes.

color package by David Carlisle provides LaTeX users with all the necessary macros which add color, create a more eye-pleasing document over a computer screen and attract the attention of the reader to a particularly important point.

hyperref package by Sebastian Rahtz enables all the LaTeX cross-referencing capabilities to be translated into hypertext links.

It is widely used for producing hyperlinks, bookmarks, thumbnails, navigation buttons and other navigation tools, and it works well with almost all LaTeX presentation packages.

Java Power Presenter - JPP is a presentation bundle, created by Thorsten Ehm (University of Augsburg, Germany), offering a platform-independent way to build PowerPoint-like presentations in LaTeX reusing all the previously written LaTeX code.

It consist of the LaTeX2ε class file (jpp.cls) based on the LaTeX Seminar style, and a software written in Java.

color package by David Carlisle provides LaTeX users with all the necessary macros which add color, create a more eye-pleasing document over a computer screen and attract the attention of the reader to a particularly important point.

hyperref package by Sebastian Rahtz enables all the LaTeX cross-referencing capabilities to be translated into hypertext links.

It is widely used for producing hyperlinks, bookmarks, thumbnails, navigation buttons and other navigation tools, and it works well with almost all LaTeX presentation packages.

Java Power Presenter - JPP is a presentation bundle, created by Thorsten Ehm (University of Augsburg, Germany), offering a platform-independent way to build PowerPoint-like presentations in LaTeX reusing all the previously written LaTeX code.

It consist of the LaTeX2ε class file (jpp.cls) based on the LaTeX Seminar style, and a software written in Java.

color package by David Carlisle provides LaTeX users with all the necessary macros which add color, create a more eye-pleasing document over a computer screen and attract the attention of the reader to a particularly important point.

hyperref package by Sebastian Rahtz enables all the LaTeX cross-referencing capabilities to be translated into hypertext links.

It is widely used for producing hyperlinks, bookmarks, thumbnails, navigation buttons and other navigation tools, and it works well with almost all LaTeX presentation packages.

Java Power Presenter - JPP is a presentation bundle, created by Thorsten Ehm (University of Augsburg, Germany), offering a platform-independent way to build PowerPoint-like presentations in LaTeX reusing all the previously written LaTeX code.

It consist of the LaTeX2ε class file (jpp.cls) based on the LaTeX Seminar style, and a software written in Java.

color package by David Carlisle provides LaTeX users with all the necessary macros which add color, create a more eye-pleasing document over a computer screen and attract the attention of the reader to a particularly important point.

hyperref package by Sebastian Rahtz enables all the LaTeX cross-referencing capabilities to be translated into hypertext links.

It is widely used for producing hyperlinks, bookmarks, thumbnails, navigation buttons and other navigation tools, and it works well with almost all LaTeX presentation packages.

Java Power Presenter - JPP is a presentation bundle, created by Thorsten Ehm (University of Augsburg, Germany), offering a platform-independent way to build PowerPoint-like presentations in LaTeX reusing all the previously written LaTeX code.

It consist of the LaTeX2ε class file (jpp.cls) based on the LaTeX Seminar style, and a software written in Java.

color package by David Carlisle provides LaTeX users with all the necessary macros which add color, create a more eye-pleasing document over a computer screen and attract the attention of the reader to a particularly important point.

hyperref package by Sebastian Rahtz enables all the LaTeX cross-referencing capabilities to be translated into hypertext links.

It is widely used for producing hyperlinks, bookmarks, thumbnails, navigation buttons and other navigation tools, and it works well with almost all LaTeX presentation packages.

Java Power Presenter - JPP is a presentation bundle, created by Thorsten Ehm (University of Augsburg, Germany), offering a platform-independent way to build PowerPoint-like presentations in LaTeX reusing all the previously written LaTeX code.

It consist of the LaTeX2ε class file (jpp.cls) based on the LaTeX Seminar style, and a software written in Java.

Utopia PDF Presentations Bundle provides accessories for production of excellent PowerPoint-like presentations from a LaTeX source.

The process requires generation of PDF by way of PostScript, as some effects (notably "builds/incremental display") are implemented by post-processing the PostScript generated from TeX.

The bundle consists of two LaTeX packages, working harmoniously with most document classes, and the PostScript post-processor.

This is the only commercial program among the mentioned ones.

Utopia PDF Presentations Bundle provides accessories for production of excellent PowerPoint-like presentations from a LaTeX source.

The process requires generation of PDF by way of PostScript, as some effects (notably "builds/incremental display") are implemented by post-processing the PostScript generated from TeX.

The bundle consists of two LaTeX packages, working harmoniously with most document classes, and the PostScript post-processor.

This is the only commercial program among the mentioned ones.

Utopia PDF Presentations Bundle provides accessories for production of excellent PowerPoint-like presentations from a LaTeX source.

The process requires generation of PDF by way of PostScript, as some effects (notably "builds/incremental display") are implemented by post-processing the PostScript generated from TeX.

The bundle consists of two LaTeX packages, working harmoniously with most document classes, and the PostScript post-processor.

This is the only commercial program among the mentioned ones.

Utopia PDF Presentations Bundle provides accessories for production of excellent PowerPoint-like presentations from a LaTeX source.

The process requires generation of PDF by way of PostScript, as some effects (notably "builds/incremental display") are implemented by post-processing the PostScript generated from TeX.

The bundle consists of two LaTeX packages, working harmoniously with most document classes, and the PostScript post-processor.

This is the only commercial program among the mentioned ones.

PPower4 – PDF Presentation Post Processor

PPower4 is a post processor of PDF documents written by Klaus Guntermann (Technical University of Darmstadt, Germany).

It is used to post process presentations in PDF format which were prepared using LaTeX to add dynamic and background effects.

PPower4 can only provide those effects, which are implemented in the reader.

The PDF files can be created with pdfLaTeX or with standard LaTeX and then converted to PDF with dvipdfm.

The post processing software is written in Java and it is free. It has been run successfully with Java 1.2.x and better (Java Runtime Environment (JRE) is sufficient).

PPower4 – PDF Presentation Post Processor

PPower4 is a post processor of PDF documents written by Klaus Guntermann (Technical University of Darmstadt, Germany).

It is used to post process presentations in PDF format which were prepared using LaTeX to add dynamic and background effects.

PPower4 can only provide those effects, which are implemented in the reader.

The PDF files can be created with pdfLaTeX or with standard LaTeX and then converted to PDF with dvipdfm.

The post processing software is written in Java and it is free. It has been run successfully with Java 1.2.x and better (Java Runtime Environment (JRE) is sufficient).

PPower4 – PDF Presentation Post Processor

PPower4 is a post processor of PDF documents written by Klaus Guntermann (Technical University of Darmstadt, Germany).

It is used to post process presentations in PDF format which were prepared using LaTeX to add dynamic and background effects.

PPower4 can only provide those effects, which are implemented in the reader.

The PDF files can be created with pdfLaTeX or with standard LaTeX and then converted to PDF with dvipdfm.

The post processing software is written in Java and it is free. It has been run successfully with Java 1.2.x and better (Java Runtime Environment (JRE) is sufficient).

PPower4 – PDF Presentation Post Processor

PPower4 is a post processor of PDF documents written by Klaus Guntermann (Technical University of Darmstadt, Germany).

It is used to post process presentations in PDF format which were prepared using LaTeX to add dynamic and background effects.

PPower4 can only provide those effects, which are implemented in the reader.

The PDF files can be created with pdfLaTeX or with standard LaTeX and then converted to PDF with dvipdfm.

The post processing software is written in Java and it is free. It has been run successfully with Java 1.2.x and better (Java Runtime Environment (JRE) is sufficient).

PPower4 – PDF Presentation Post Processor

PPower4 is a post processor of PDF documents written by Klaus Guntermann (Technical University of Darmstadt, Germany).

It is used to post process presentations in PDF format which were prepared using LaTeX to add dynamic and background effects.

PPower4 can only provide those effects, which are implemented in the reader.

The PDF files can be created with pdfLaTeX or with standard LaTeX and then converted to PDF with dvipdfm.

The post processing software is written in Java and it is free. It has been run successfully with Java 1.2.x and better (Java Runtime Environment (JRE) is sufficient).

PPower4 provides a small LaTeX package (pause.sty) which let's the user insert small colored spots (using the command `\pause`) in the PDF file where a break should be make during display.

During postprocessing PPower4 removes these colored chunks and adjusts the page number. This makes an impression that the same page is displayed step by step.

Additional packages are provided for setting

- background colors (background.sty) and
- page transitions (pagetrans.tex)

The last is actually a feature of hyperref.sty and can be used with any LaTeX based solution.

PPower4 provides a small LaTeX package (pause.sty) which let's the user insert small colored spots (using the command `\pause`) in the PDF file where a break should be make during display.

During postprocessing PPower4 removes these colored chunks and adjusts the page number. This makes an impression that the same page is displayed step by step.

Additional packages are provided for setting

- background colors (background.sty) and
- page transitions (pagetrans.tex)

The last is actually a feature of hyperref.sty and can be used with any LaTeX based solution.

PPower4 provides a small LaTeX package (pause.sty) which let's the user insert small colored spots (using the command `\pause`) in the PDF file where a break should be make during display.

During postprocessing PPower4 removes these colored chunks and adjusts the page number. This makes an impression that the same page is displayed step by step.

Additional packages are provided for setting

- **background colors (background.sty) and**
- **page transitions (pagetrans.tex)**

The last is actually a feature of hyperref.sty and can be used with any LaTeX based solution.

PPower4 provides a small LaTeX package (pause.sty) which let's the user insert small colored spots (using the command `\pause`) in the PDF file where a break should be make during display.

During postprocessing PPower4 removes these colored chunks and adjusts the page number. This makes an impression that the same page is displayed step by step.

Additional packages are provided for setting

- background colors (background.sty) and
- page transitions (pagetrans.tex)

The last is actually a feature of hyperref.sty and can be used with any LaTeX based solution.

Example 1: Table builds

Example 1: Table builds

Building	your	table

Example 1: Table builds

Building	your	table
line	by	line,

Example 1: Table builds

Building	your	table
line	by	line,
entry		

Example 1: Table builds

Building	your	table
line	by	line,
entry	**by**	

Example 1: Table builds

Building	your	table
line	by	line,
entry	by	**entry,**

Example 1: Table builds

Building	your	table
line	by	line,
entry	by	entry,

growing	up,	too.

Example 1: Table builds

Building	your	table
line	by	line,
entry	by	entry,
from	**the**	**bottom**
growing	up,	too.

Example 2: Building tables by columns

Example 2: Building tables by columns

Show
the
first
column,

Example 2: Building tables by columns

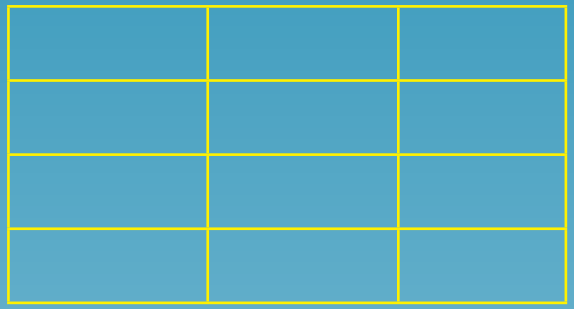
Show	then
the	the
first	second,
column,	and

Example 2: Building tables by columns

Show	then	finally,
the	the	also
first	second,	the
column,	and	third.

Example 3: Filling tables by columns

Example 3: Filling tables by columns



Example 3: Filling tables by columns

Fill		
the		
first		
column,		

Example 3: Filling tables by columns

Fill	then	
the	the	
first	second,	
column,	and	

Example 3: Filling tables by columns

Fill	then	finally,
the	the	also
first	second,	the
column,	and	third.

Example 4: More filled tables

Example 4: More filled tables

Example 4: More filled tables

Fill		
	the	
		table

Example 4: More filled tables

Fill		at
	the	
		table

Example 4: More filled tables

Fill		at
	the	
random		
	positions	table

Example 4: More filled tables

Fill	leave	at
some	the	
random		empty
	positions	table

Example 4: More filled tables

Fill	leave	
some	the	well,
random	almost	empty
empty	positions	table

Example 4: More filled tables

Fill	leave	
some	the	well,
random	almost	empty
empty	positions	table

Did you notice, that the element in the upper right corner has vanished?

TeXPower

TeXPower is a bundle of style and class files for creating dynamic online presentations with LaTeX written by Stephan Lehmke (University of Dortmund, Germany).

TeXPower

TeXPower is a bundle of style and class files for creating dynamic online presentations with LaTeX written by Stephan Lehmke (University of Dortmund, Germany).

TeXPower is not a complete presentation package – it is an add-on to other document classes which just adds dynamic presentation effects (and some other gimmicks specifically interesting for dynamic presentations).

TeXPower

TeXPower is a bundle of style and class files for creating dynamic online presentations with LaTeX written by Stephan Lehmke (University of Dortmund, Germany).

TeXPower is not a complete presentation package – it is an add-on to other document classes which just adds dynamic presentation effects (and some other gimmicks specifically interesting for dynamic presentations).

It works well in conjunction with pdfslide and pdfscreen, as well as with seminar, foils, or any other class/package for designing slides.

TeXPower is meant as an alternative to PPower4 for those who can not use pdfLaTeX (for instance, because they're using PSTricks), because it can also be used with tex → dvi → ps → pdf workflow.

TeXPower is meant as an alternative to PPower4 for those who can not use pdfLaTeX (for instance, because they're using PSTricks), because it can also be used with tex $\longrightarrow$ dvi $\longrightarrow$ ps $\longrightarrow$ pdf workflow.

No post-processing or additional tools are needed – the standard LaTeX distribution will do.

TeXPower is meant as an alternative to PPower4 for those who can not use pdfLaTeX (for instance, because they're using PSTricks), because it can also be used with tex $\longrightarrow$ dvi $\longrightarrow$ ps $\longrightarrow$ pdf workflow.

No post-processing or additional tools are needed – the standard LaTeX distribution will do.

However, using the pp4slide.sty with TeXPower and then postprocessing the PDF presentation with PPower4, some additional effects can be achieved.

TeXPower is meant as an alternative to PPower4 for those who can not use pdfLaTeX (for instance, because they're using PSTricks), because it can also be used with tex → dvi → ps → pdf workflow.

No post-processing or additional tools are needed – the standard LaTeX distribution will do.

However, using the pp4slide.sty with TeXPower and then postprocessing the PDF presentation with PPower4, some additional effects can be achieved.

The heart of the bundle is the TeXPower package (texpower.sty), which enables some commands for dynamic presentation effects in PostScript and PDF documents produced from LaTeX.

TeXPower is meant as an alternative to PPower4 for those who can not use pdfLaTeX (for instance, because they're using PSTricks), because it can also be used with tex → dvi → ps → pdf workflow.

No post-processing or additional tools are needed – the standard LaTeX distribution will do.

However, using the pp4slide.sty with TeXPower and then postprocessing the PDF presentation with PPower4, some additional effects can be achieved.

The heart of the bundle is the TeXPower package (texpower.sty), which enables some commands for dynamic presentation effects in PostScript and PDF documents produced from LaTeX.

This includes page transitions, color highlighting and displaying pages incrementally.

The AcroTEX eDucation Bundle

The AcroTEX eDucation Bundle is a collection of LATEX macro files, along with various support and sample files, created by D. P. Story (University of Akron, USA).

The overall theme of the bundle is ePublication in the education sector using LATEX as the authoring application and Adobes Portable Document Format (PDF) as the file format of the output document.

It should be useful to educators who want to post interactive materials for their students on the web.

The AcroTEX eDucation Bundle

The AcroTEX eDucation Bundle is a collection of LaTeX macro files, along with various support and sample files, created by D. P. Story (University of Akron, USA).

The overall theme of the bundle is ePublication in the education sector using LaTeX as the authoring application and Adobes Portable Document Format (PDF) as the file format of the output document.

It should be useful to educators who want to post interactive materials for their students on the web.

The AcroTEX eDucation Bundle

The AcroTEX eDucation Bundle is a collection of LATEX macro files, along with various support and sample files, created by D. P. Story (University of Akron, USA).

The overall theme of the bundle is ePublication in the education sector using LATEX as the authoring application and Adobes Portable Document Format (PDF) as the file format of the output document.

It should be useful to educators who want to post interactive materials for their students on the web.

The AcroTEX eDucation Bundle has four basic components:

web package is used to create an attractive, easy-on-the-eye page layout suitable for the web or classroom/conference presentations;

exerquiz package makes it very easy to create interactive exercises and quizzes;

insdljs package allows for the automatics insertion of document level JavaScript, which is used for processing of the exerquiz quizzes;

dljslib package is used as a library of JavaScript functions.

The AcroTEX eDucation Bundle has four basic components:

web package is used to create an attractive, easy-on-the-eye page layout suitable for the web or classroom/conference presentations;

exerquiz package makes it very easy to create interactive exercises and quizzes;

insdljs package allows for the automatics insertion of document level JavaScript, which is used for processing of the exerquiz quizzes;

dljslib package is used as a library of JavaScript functions.

The AcroTEX eDucation Bundle has four basic components:

web package is used to create an attractive, easy-on-the-eye page layout suitable for the web or classroom/conference presentations;

exerquiz package makes it very easy to create interactive exercises and quizzes;

insdljs package allows for the automatics insertion of document level JavaScript, which is used for processing of the exerquiz quizzes;

dljslib package is used as a library of JavaScript functions.

The AcroTEX eDucation Bundle has four basic components:

web package is used to create an attractive, easy-on-the-eye page layout suitable for the web or classroom/conference presentations;

exerquiz package makes it very easy to create interactive exercises and quizzes;

insdljs package allows for the automatics insertion of document level JavaScript, which is used for processing of the exerquiz quizzes;

dljslib package is used as a library of JavaScript functions.

The AcroTEX eDucation Bundle has four basic components:

web package is used to create an attractive, easy-on-the-eye page layout suitable for the web or classroom/conference presentations;

exerquiz package makes it very easy to create interactive exercises and quizzes;

insdljs package allows for the automatics insertion of document level JavaScript, which is used for processing of the exerquiz quizzes;

dljslib package is used as a library of JavaScript functions.

There are also two additional components:

eq2db package is used to customize exerquiz to save results of the quiz environment to a database.

It has three options

- **eqRecord** – a simple ASP script that takes some exerquiz data and saves it to a database;
- **eqEmail** – an ASP script that e-mails quiz results to the instructor;
- **custom** – this is a hook for other developers to use this package and its macros.

eForm support defines six basic (and internal) commands for creating the six types of form elements used in the AcroTEX quizzes – push buttons, check boxes, radio buttons, list boxes, combo boxes and text fields.

There are also two additional components:

eq2db package is used to customize exerquiz to save results of the quiz environment to a database.

It has three options

- **eqRecord** – a simple ASP script that takes some exerquiz data and saves it to a database;
- **eqEmail** – an ASP script that e-mails quiz results to the instructor;
- **custom** – this is a hook for other developers to use this package and its macros.

eForm support defines six basic (and internal) commands for creating the six types of form elements used in the AcroTEX quizzes – push buttons, check boxes, radio buttons, list boxes, combo boxes and text fields.

There are also two additional components:

eq2db package is used to customize exerquiz to save results of the quiz environment to a database.

It has three options

eqRecord – a simple ASP script that takes some exerquiz data and saves it to a database;

eqEmail – an ASP script that e-mails quiz results to the instructor;

custom – this is a hook for other developers to use this package and its macros.

eForm support defines six basic (and internal) commands for creating the six types of form elements used in the AcroTEX quizzes – push buttons, check boxes, radio buttons, list boxes, combo boxes and text fields.

There are also two additional components:

eq2db package is used to customize exerquiz to save results of the quiz environment to a database.

It has three options

eqRecord – a simple ASP script that takes some exerquiz data and saves it to a database;

eqEmail – an ASP script that e-mails quiz results to the instructor;

custom – this is a hook for other developers to use this package and its macros.

eForm support defines six basic (and internal) commands for creating the six types of form elements used in the AcroTEX quizzes – push buttons, check boxes, radio buttons, list boxes, combo boxes and text fields.

There are also two additional components:

eq2db package is used to customize exerquiz to save results of the quiz environment to a database.

It has three options

eqRecord – a simple ASP script that takes some exerquiz data and saves it to a database;

eqEmail – an ASP script that e-mails quiz results to the instructor;

custom – this is a hook for other developers to use this package and its macros.

eForm support defines six basic (and internal) commands for creating the six types of form elements used in the AcroTEX quizzes – push buttons, check boxes, radio buttons, list boxes, combo boxes and text fields.

There are also two additional components:

eq2db package is used to customize exerquiz to save results of the quiz environment to a database.

It has three options

eqRecord – a simple ASP script that takes some exerquiz data and saves it to a database;

eqEmail – an ASP script that e-mails quiz results to the instructor;

custom – this is a hook for other developers to use this package and its macros.

eForm support defines six basic (and internal) commands for creating the six types of form elements used in the AcroTEX quizzes – push buttons, check boxes, radio buttons, list boxes, combo boxes and text fields.

There are also two additional components:

eq2db package is used to customize exerquiz to save results of the quiz environment to a database.

It has three options

eqRecord – a simple ASP script that takes some exerquiz data and saves it to a database;

eqEmail – an ASP script that e-mails quiz results to the instructor;

custom – this is a hook for other developers to use this package and its macros.

eForm support defines six basic (and internal) commands for creating the six types of form elements used in the AcroTEX quizzes – push buttons, check boxes, radio buttons, list boxes, combo boxes and text fields.

There are also two additional components:

eq2db package is used to customize exerquiz to save results of the quiz environment to a database.

It has three options

eqRecord – a simple ASP script that takes some exerquiz data and saves it to a database;

eqEmail – an ASP script that e-mails quiz results to the instructor;

custom – this is a hook for other developers to use this package and its macros.

eForm support defines six basic (and internal) commands for creating the six types of form elements used in the AcroTEX quizzes – push buttons, check boxes, radio buttons, list boxes, combo boxes and text fields.

AcroTEX Examples