

Implementasi PDFx dan PDFtk dalam Pembuatan Berkas PDF DIPA Petikan

Didik Setiawan
di2k.setiawan@gmail.com

Lisensi Dokumen:

Copyright © IlmuKomputer.Com

Seluruh dokumen di IlmuKomputer.Com dapat digunakan, dimodifikasi dan disebarkan secara bebas untuk tujuan bukan komersial (nonprofit), dengan syarat tidak menghapus atau merubah atribut penulis dan pernyataan copyright yang disertakan dalam setiap dokumen. Tidak diperbolehkan melakukan penulisan ulang, kecuali mendapatkan ijin terlebih dahulu dari IlmuKomputer.Com.

DIPA merupakan dokumen pelaksanaan anggaran yang berfungsi sebagai dokumen perencanaan, pelaksanaan, pengendalian/pengawasan, evaluasi/pelaporan, serta dokumen pendukung kegiatan akuntansi pemerintah. DIPA terdiri atas DIPA Induk dan DIPA Petikan.

Sejak tahun anggaran 2013 DIPA Petikan yang memuat alokasi anggaran untuk masing-masing Satuan Kerja (Satker) tidak lagi menggunakan tanda tangan basah sebagai bentuk pengesahan (otentifikasi). Keabsahan DIPA Petikan dijamin melalui sistem dengan penggunaan kode pengaman berupa *digital stamp* sebagai pengganti tanda tangan pengesahan.

Dengan digunakannya *digital stamp* sebagai pengganti otentifikasi maka pencetakan *hard copy* DIPA Petikan hanya dilakukan dalam rangka *ceremonial* penyerahan DIPA awal, selebihnya distribusi DIPA awal dilakukan dalam bentuk *softcopy*.

Pada proses revisi, sistem akan membuat dan mendistribusikan DIPA Petikan dalam bentuk *softcopy*. Untuk menjamin konsistensi hasil cetakan DIPA Petikan maka distribusi *soft copy* DIPA Petikan dilakukan dengan menggunakan format berkas PDF.

Tulisan ini berisi metode pembuatan berkas PDF DIPA Petikan pada bahasa pemrograman Microsoft Visual FoxPro dengan menggunakan *tool* PDFx dan PDFtk.

DIPA Petikan

Daftar Isian Pelaksanaan Anggaran (DIPA) sebagai dokumen pelaksanaan anggaran disusun oleh Pengguna Anggaran (PA)/Kuasa Pengguna Anggaran (KPA) dan disahkan oleh Direktur Jenderal Anggaran (DJA) atas nama Menteri Keuangan.

DIPA Induk adalah akumulasi dari DIPA Satker yang disusun oleh PA menurut Unit Eselon I Kementerian Negara/Lembaga. DIPA Petikan adalah DIPA per Satker yang dicetak secara otomatis melalui sistem yang dilengkapi dengan kode pengaman berupa *digital stamp* sebagai pengganti tanda tangan pengesahan (otentifikasi) .

DIPA Petikan disusun menggunakan data yang berasal dari Rencana Kerja Anggaran (RKA) Satker yang telah disesuaikan dengan Alokasi Anggaran Kementerian Negara/Lembaga dan mendapat persetujuan DPR, telah ditelaah antara Kementerian Negara/Lembaga, Kementerian Perencanaan Pembangunan Nasional (PPN) dan Kementerian Keuangan c.q. DJA serta telah ditetapkan dalam Keputusan Presiden (Keppres) mengenai Rincian Anggaran Belanja Pemerintah Pusat (RABPP).

DIPA Petikan digunakan sebagai dasar pelaksanaan kegiatan Satker dan pencairan dana/pengesahan bagi Bendahara Umum Negara (BUN)/Kuasa Bendahara Umum Negara (Kuasa BUN) yang merupakan kesatuan yang tidak terpisahkan dari DIPA Induk.

DIPA Petikan terdiri atas :

1. Lembar Surat Pengesahan DIPA Petikan (SP DIPA Petikan);
2. Halaman I memuat informasi Kinerja dan Sumber Dana yang terdiri dari :
 - a. halaman I A mengenai informasi Kinerja;
 - b. halaman I B mengenai Sumber Dana.
3. Halaman II memuat Rincian Pengeluaran;
4. Halaman III memuat Rencana Penarikan Dana dan Perkiraan Penerimaan; dan
5. Halaman IV memuat Catatan

DIPA Petikan yang telah disahkan oleh DJA atas nama Menteri Keuangan digandakan dan disampaikan kepada :

1. Satker bersangkutan;
2. Kepala Kantor Pelayanan Perbendaharaan Negara pembayar;
3. Kepala Kanwil Direktorat Jenderal Perbendaharaan;
4. Direktur Jenderal Anggaran c.q. Direktur Anggaran I,II, dan III;
5. Menteri/Pimpinan Lembaga :
 - a. Sekretaris Jenderal;
 - b. Inspektur Jenderal;
 - c. Pimpinan Unit Eselon I penanggung jawab Program;
6. Ketua Badan Pemeriksa Keuangan;
7. Gubernur;
8. Direktur Jenderal Perbendaharaan :
 - a. Direktur Pelaksana Anggaran;
 - b. Direktur Akuntansi dan Pelaporan Keuangan.

Penyampaian DIPA Petikan untuk butir 1 s.d. 4 berupa *hardcopy*, sedangkan untuk butir 5 s.d. 8 berupa *softcopy*. Penyampaian *hardcopy* DIPA Petikan (butir 1 s.d. 4) hanya dilakukan untuk DIPA Petikan awal yang diterbitkan sebelum tahun anggaran berjalan.

DIPA Petikan awal dan atau DIPA Petikan hasil revisi pada tahun anggaran berjalan diterbitkan dalam bentuk *softcopy* dan diperlakukan sebagaimana dokumen *hardcopy* DIPA Petikan. Satker dapat melakukan pencetakan DIPA Petikan secara mandiri dengan cara mengunduh *softcopy* DIPA Petikan yang telah disetujui dan didistribusikan secara online untuk selanjutnya dipergunakan sebagai dasar pelaksanaan kegiatan.

Kode pengaman berupa *digital stamp* digunakan untuk menjamin kebenaran dan kesesuaian antara Arsip Data Komputer (ADK) berupa backup data RKAKL DIPA dengan *softcopy* DIPA Petikan dan untuk menjamin konsistensi hasil cetakan DIPA Petikan maka distribusi *softcopy* DIPA Petikan dilakukan dengan menggunakan format berkas PDF.

PDF

PDF (*Portable Document Format*) adalah sebuah format berkas (*file*) yang dibuat oleh Adobe Systems pada tahun 1993 untuk keperluan pertukaran dokumen digital. PDF telah menjadi standar ISO (*International Organization for Standardization*) pada tanggal 1 Juli 2008 dengan kode ISO 32000-1:2008.

Antarmuka dokumen PDF pada umumnya tersusun atas kombinasi teks, grafik vektor, dan grafik raster. Grafik vektor digunakan untuk menampilkan ilustrasi yang terbentuk dari garis dan kurva, sedangkan grafik raster digunakan untuk menampilkan foto dan citra.

Dokumen PDF mendukung *hyperlink*, *forms*, *JavaScript*, dan berbagai kemampuan lain dengan penambahan *plugin*.

Berikut adalah beberapa nilai lebih penggunaan format PDF :

1. Cepat dan mudah dibuat, dirancang untuk memenuhi tuntutan dunia kerja yang serba cepat.
Saat ini, hampir semua dokumen bisa dijadikan berkas PDF;
2. Aman, dapat dienkripsi, diberi tanda air (*watermark*) untuk memastikan bahwa berkas tersebut tidak mengalami perubahan pada saat dikirim;
3. Jaminan Keaslian, dapat diberi sertifikat digital dan atau tanda tangan digital untuk menjamin keasliannya;
4. *Portabel*, mudah dipindahkan, dikenal dan dapat dibaca oleh semua sistem operasi (Mac OS, Windows, LINUX, UNIX, Android, dll). Ukurannya lebih kecil dari jenis dokumen yang lain. Berkas PDF adalah dokumen yang paling efisien untuk diunduh dan diunggah;
5. Terkenal, dimana berkas PDF adalah salah satu bentuk buku elektronik yang paling populer. Banyak tersedia perangkat lunak untuk membuat dan membaca berkas PDF, baik yang berbayar maupun yang cuma-cuma. Tampilan berkas PDF mirip dengan kertas biasa. Sehingga menarik, mudah dipahami dan disukai;
6. Akurat, *WYSIWYG* (*What You See Is What You Get*) apa yang terlihat di layar komputer akan persis sama dengan hasil cetakan.

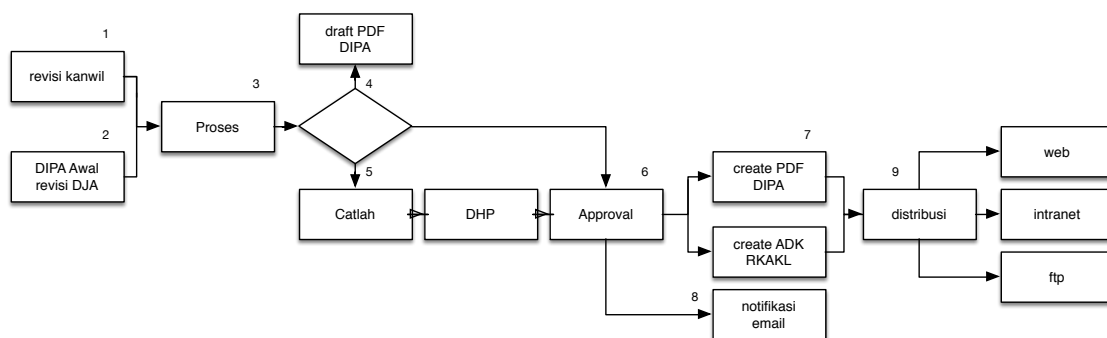
Berdasarkan karakteristik tersebut diatas maka *softcopy* dokumen DIPA Petikan didistribusikan dengan menggunakan format PDF.

Pembuatan *softcopy* DIPA Petikan dalam format PDF merupakan bagian dari alur proses penyelesaian DIPA awal yang melibatkan DJA maupun proses revisi anggaran yang melibatkan DJA dan Kantor Wilayah (Kanwil) Direktorat Jenderal Perbendaharaan (DJPbn).

Alur Proses

DJA dalam penyelesaian DIPA awal dan proses revisi kewenangan DJA menggunakan sistem Aplikasi DJA Single Window (DSW) sedangkan Kanwil DJPbn menggunakan sistem Aplikasi Revisi untuk penyelesaian proses revisi anggaran kewenangan Kanwil.

Sistem Aplikasi DSW dan Aplikasi Revisi berfungsi sebagai antar muka (*interface*) yang menghubungkan user dengan sistem Proses DIPA. Sistem Proses DIPA merupakan sistem terpusat yang berfungsi melakukan proses penyelesaian DIPA secara otomatis berdasarkan permintaan proses (*request*) yang berasal dari DJA (Aplikasi DSW) maupun Kanwil (Aplikasi Revisi). Secara sederhana alur proses yang terdapat pada sistem Proses DIPA adalah sebagai berikut :



Bagan 1. Alur Proses DIPA Petikan

Revisi Kanwil⁽¹⁾ diajukan oleh Satker melalui Kanwil dimana Satker tersebut berada. Sistem Aplikasi Revisi akan melakukan pengecekan kewenangan usulan revisi, jika usulan revisi berada dalam kewenangan Kanwil dan tidak ada proses revisi untuk satker tersebut di

DJA maka sistem akan mengirimkan permintaan proses (*request*) ke server proses DIPA beserta ADK yang telah diunggah dan melakukan update status revisi.

DIPA awal atau revisi DJA⁽²⁾ diajukan oleh Unit Eselon I melalui DJA. Untuk proses revisi, Sistem Aplikasi DSW akan melakukan pengecekan apakah terdapat Satker dalam usulan revisi tersebut yang juga sedang melakukan revisi di Kanwil. Apabila terdapat revisi di Kanwil maka otomatis sistem akan membatalkan proses revisi Kanwil, selanjutnya sistem akan mengirimkan permintaan proses (*request*) ke server proses DIPA disertai ADK yang telah diunggah dan melakukan update status revisi.

Blok proses⁽³⁾ (Sistem proses DIPA) menerima dan mengatur proses yang akan dilalui selanjutnya berdasarkan permintaan proses dari Kanwil⁽¹⁾ dan DJA⁽²⁾. Dalam blok proses⁽³⁾ ADK usulan revisi (dan DIPA awal) akan di proses ke database *temporary* sesuai dengan asal usulan. Database *temporary* DJA untuk usulan DJA, database *temporary* Kanwil untuk usulan Kanwil.

Dalam hal permintaan proses (*request*) berupa usulan revisi baik dari DJA maupun Kanwil maka sistem akan melakukan proses pembentukan berkas PDF konsep (*draft*) DIPA⁽⁴⁾ dengan mengacu pada database usulan.

Apabila konsep (*draft*) DIPA⁽⁴⁾ telah disetujui maka user dapat melakukan tahapan persetujuan (*approval*)⁽⁶⁾ sebagai tahap pengesahan dan menerbitkan DIPA Petikan hasil revisi.

Dalam hal proses berasal dari DJA⁽⁵⁾ maka diperlukan tahapan proses pembuatan Catatan Penelaahan (Catlah) dan pembuatan Daftar Hasil Penelaahan (DHP) sebelum melakukan proses *approval*⁽⁶⁾.

Proses *approval*⁽⁶⁾ yang dilakukan DJA atau Kanwil memerintahkan sistem untuk melakukan proses pembentukan berkas PDF DIPA Petikan⁽⁷⁾ dan proses pembentukan ADK backup data RKAKL, selanjutnya sistem mengirimkan pemberitahuan (notifikasi) melalui surat elektronik (*email*)⁽⁸⁾ yang berisi informasi penyelesaian proses DIPA kepada para pemangku kepentingan (*stakeholder*).

Distribusi⁽⁹⁾ *softcopy* dokumen DIPA Petikan beserta ADK RKAKL dilakukan secara otomatis setelah proses PDF dan ADK selesai untuk menjamin user dapat memperoleh *softcopy* dokumen DIPA beserta ADK RKAKL yang telah disetujui pada saat itu juga. Distribusi dilakukan melalui jalur internet (website) untuk kebutuhan eksternal, intranet (sharing folder DJA) untuk kebutuhan internal DJA, dan ftp untuk kebutuhan internal Direktorat Jenderal Perbendaharaan (DJPbn).

Softcopy dokumen DIPA Petikan beserta ADK RKAKL dapat diunduh oleh masing-masing Satuan Kerja (Satker), Unit Eselon I, atau pihak terkait lainnya sesuai dengan kewenangan yang dimilikinya melalui alamat : <http://rkakldipa.anggaran.depkeu.go.id>.

Berdasarkan alur proses tersebut diatas tampak bahwa berkas PDF DIPA Petikan dihasilkan saat pembentukan konsep (*draft*) dan net DIPA Petikan. Konsep DIPA Petikan digunakan sebagai usulan persetujuan, tidak disertai dengan *barcode digital stamp*, dan tidak didistribusikan. Net DIPA Petikan disertai dengan *barcode digital stamp*, didistribusikan, dan berlaku sebagai dokumen yang sah.

Seluruh sistem aplikasi *desktop* proses DIPA dibangun dengan menggunakan bahasa pemrograman Microsoft Visual FoxPro versi 9.0 dan sistem RKAKL DIPA online dibangun dengan menggunakan bahasa pemrograman PHP.

Untuk menghasilkan berkas dokumen DIPA Petikan yang terdiri dari beberapa format halaman DIPA dengan menggunakan bahasa pemrograman Visual FoxPro, penulis menggunakan *tools* sebagai berikut :

PDFx

PDFx digunakan sebagai *PDF Report Listener* yang berfungsi untuk meng-*generate report* kedalam format PDF.

PDFx merupakan *class* Visual FoxPro yang dikembangkan oleh Luis Guillermo Navas dengan merujuk pada HARU *library*. PDFx berfungsi sebagai *PDF Report Listener* untuk

menghasilkan berkas format PDF yang bersumber dari *report* Visual FoxPro (.frx). PDFx dapat digunakan pada Visual FoxPro versi 9.0 dan bersifat *free*.

Source PDFx dapat diunduh di : <http://weblogs.foxite.com/luisnavas/default.aspx>

Untuk dapat menggunakan PDFx pada *project* Visual FoxPro lakukan langkah-langkah berikut :

1. *Include* PDFx.VCX pada *project*;
2. *Copy* System.APP (GDIPLUSX *library*) pada folder *project*;
3. *Copy* libhpdf.dll pada folder *project*;

Proses *include* PDFx pada *project* akan menyertakan *class* PDFx pada hasil *compile* aplikasi (.exe atau .app) yang didistribusikan.

Berikut contoh perintah untuk menjalankan PDFx beserta parameter yang dapat digunakan:

```
Local loListener As ReportListener

loListener = NewObject('PdfListener1', 'Libs\PDFx.vcx')
&&Use 'PdfListener2' to create PDF as an Image
loListener.cTargetFileName = "Ref_Unit.Pdf" && Output File
loListener.QuietMode=.F.
loListener.lEncryptDocument=.F.
loListener.lCanEdit=.F. &&Only if lEncryptDocument=.T.
loListener.lCanCopy=.T. &&Only if lEncryptDocument=.T.
loListener.lCanAddNotes=.F. &&Only if lEncryptDocument=.T.
loListener.lCanPrint=.T. &&Only if lEncryptDocument=.T.
loListener.cMasterPassword="masterpassword" &&Only if lEncryptDocument=.T.
loListener.cUserPassword="userpassword" &&Only if lEncryptDocument=.T.
loListener.lOpenViewer=.T.

Report Form r_tunit Object loListener
```

Perintah diatas menghasilkan berkas PDF dengan nama Ref_Unit.pdf dari *report* r_unit.frx.

PDFtk

PDFtk atau PDF *Toolkit* adalah *tool Open Source* lintas *platform* yang dikembangkan untuk melakukan perubahan (manipulasi) berkas elektronik dalam format PDF. PDFtk pada dasarnya adalah antar muka yang dikembangkan dari *library* iText (di-*compile* dengan menggunakan GCJ) dengan kemampuan *splitting*, *merging*, *encrypting*, *decrypting*, *uncompressing*, *recompressing*, dan *repairing* dokumen PDF, dapat pula digunakan untuk

melakukan perubahan *watermarks*, *metadata*, dan pengisian form PDF dengan menggunakan Data FDF (*Forms Data Format*) atau Data XFDF (*XML Form Data Format*).

A. Fitur PDFtk

Berdasarkan manual Unix (*Unix man*), PDFtk memiliki fitur sebagai berikut :

1. Menggabungkan dokumen PDF atau kompilasi halaman PDF hasil scan;
2. Memecah (*Split*) halaman PDF menjadi dokumen baru;
3. *Rotate* dokumen atau halaman;
4. *Decrypt* dan atau *Encrypt* dokumen PDF;
5. Pengisian form PDF dengan menggunakan data X/FDF dan atau form isian PDF;
6. *Generate FDF Data Stencils* dari PDF *Forms*;
7. *Background Watermark* atau *Foreground Stamp*;
8. *Report PDF Metrics*, *Bookmarks* dan *Metadata*;
9. Update atau menambah PDF *Bookmarks* atau *Metadata*;
10. Menambahkan berkas (*Attach Files*) ke halaman PDF atau dokumen PDF;
11. *Unpack Attachment* PDF;
12. Memecah dokumen PDF menjadi halaman tunggal;
13. *Uncompress* dan *Re-Compress* halaman;
14. Perbaikan (*Repair*) terhadap dokumen PDF (jika memungkinkan).

B. Instalasi

Installer PDFtk dapat di download melalui alamat : <http://www.pdflabs.com/tools/pdftk-the-pdf-toolkit/>. PDFtk untuk Linux dapat diunduh dan di install sesuai dengan *package manager* masing-masing distribusi Linux.

Windows XP, Vista, 7 dan 8

Saat tulisan ini dibuat *Installer* terkini PDFtk untuk Windows adalah : `pdftk_server-1.45-windows-setup.msi`

Hasil instalasi akan membentuk direktori (*default*) di :

C:\Program Files(x86)\PDF Labs\PDFtk Server\ dengan direktori bin berisi program pdftk.exe dan direktori docs berisi manual PDFtk.

PDFtk dapat dijalankan melalui *command prompt* dengan perintah : C:\Program Files (x86)\PDF Labs\PDFtk Server\bin\pdftk.exe

Pdftk.exe merupakan file *executable* yang dapat dijalankan secara langsung pada tempat dimana file tersebut berada.

Mac OS X 10.6 (Snow Leopard), 10.7 (Lion) and 10.8 (Mountain Lion)

Saat tulisan ini dibuat *Installer* terkini PDFtk untuk Mac adalah : pdftk_server-1.45-mac_osx-10.6-setup.pkg

Hasil instalasi akan membentuk direktori (*default*) di : /opt/pdflabs/pdftk/ dengan direktori bin berisi program pdftk dan direktori docs berisi manual PDFtk.

PDFtk dapat dijalankan melalui *terminal* dengan perintah : pdftk

Linux

Instalasi di mesin Linux disesuaikan dengan *package manager* masing-masing distribusi, contoh untuk distribusi (*distro*) Debian dapat menggunakan perintah :

```
# apt-get install pdftk
```

C. PDFtk Command Line

pdftk.exe merupakan file *executable* yang dapat berjalan secara mandiri tanpa memerlukan dukungan *library* lain. Pada proses DIPA Petikan pdftk.exe diletakkan pada sub folder \pdf\, lokasi dimana berkas-berkas PDF akan digabungkan.

PDFtk merupakan program *command line* yang berjalan pada *command prompt* atau *terminal*.

Berdasarkan manual PDFtk, struktur perintah yang dapat digunakan adalah sebagai berikut :

```
13th:~ user$ pdftk
SYNOPSIS
    pdftk <input PDF files | - | PROMPT>
        [ input_pw <input PDF owner passwords | PROMPT> ]
        [ <operation> <operation arguments> ]
        [ output <output filename | - | PROMPT> ]
        [ encrypt_40bit | encrypt_128bit ]
        [ allow <permissions> ]
        [ owner_pw <owner password | PROMPT> ]
        [ user_pw <user password | PROMPT> ]
        [ flatten ] [ compress | uncompress ]
        [ keep_first_id | keep_final_id ] [ drop_xfa ]
        [ verbose ] [ dont_ask | do_ask ]
Where:
    <operation> may be empty, or:
    [ cat | shuffle | burst |
      generate_fdf | fill_form |
      background | multibackground |
      stamp | multistamp |
      dump_data | dump_data_utf8 |
      dump_data_fields | dump_data_fields_utf8 |
      update_info | update_info_utf8 |
      attach_files | unpack_files ]

For Complete Help: pdftk --help
```

Struktur perintah PDFtk pada terminal Mac OS.

Untuk menampilkan bantuan (*help*) dapat menggunakan perintah `--help` atau `-h` (`pdftk -h`)

Berikut beberapa contoh penggunaan PDFtk *command line* :

- a. Memecah dokumen PDF menjadi satu halaman per berkas :

```
$ pdftk berkas_sumber.pdf burst
```

perintah diatas akan menghasilkan berkas dengan nama `pg_0001.pdf`, `pg_0002.pdf`, dan seterusnya dari sebuah berkas dengan nama `berkas_sumber.pdf`.

Jika berkas yang ingin dihasilkan akan diberi nama dapat menggunakan perintah berikut :

```
$ pdftk berkas_sumber.pdf burst output hasil_%02d.pdf
```

perintah diatas akan menghasilkan berkas dengan nama `hasil_01.pdf`, `hasil_02.pdf`, ...

- b. Mengambil halaman tertentu pada sebuah dokumen PDF, misalnya halaman 7 sampai dengan halaman 13 dari sebuah berkas PDF dengan nama `berkas_sumber.pdf` :

```
$ pdftk berkas_sumber.pdf cat 7-13 output Hasil_hal7sd13.pdf
```

perintah diatas akan menghasilkan berkas baru dengan nama `Hasil_hal7sd13.pdf` yang berisi dokumen halaman 7 sampai dengan halaman 13 dari `berkas_sumber.pdf`.

- c. Menggabungkan beberapa berkas menjadi satu :

```
$ pdftk berkas_03.pdf berkas_04.pdf cat output gabungan.pdf
```

perintah diatas akan menggabungkan berkas_03.pdf dan berkas_04.pdf menjadi berkas baru dengan nama gabungan.pdf.

d. Menggabungkan semua berkas menjadi satu dalam sebuah direktori :

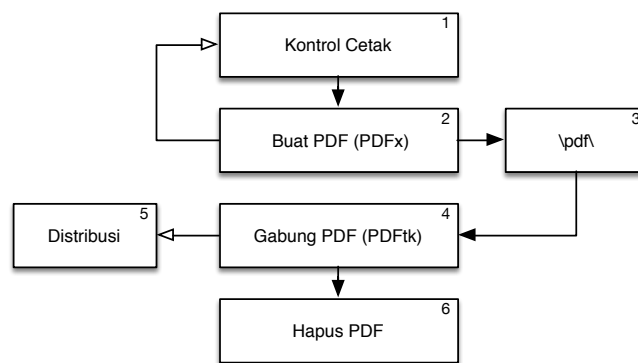
```
$ pdftk *.pdf cat output gabungan.pdf
```

perintah diatas akan menggabungkan seluruh berkas pada direktori menjadi berkas baru dengan nama gabungan.pdf.

Fungsi penggabungan berkas inilah yang digunakan untuk menyatukan berkas PDF hasil *Report Listener* PDFx untuk setiap halaman DIPA Petikan (SP, 1a, 1b, 2, 3, dan 4) menjadi satu berkas DIPA Petikan.

Implementasi

Secara sederhana alur proses pembuatan berkas PDF DIPA Petikan adalah sebagai berikut :



Bagan 2. Alur proses pembuatan berkas PDF DIPA Petikan

Kontrol cetak⁽¹⁾ berfungsi mengatur halaman DIPA yang akan diproses, fungsi ini dilakukan oleh program `p_antri_petikan.prp` didalamnya termasuk *procedure* yang akan menjalankan program `p_petikan.prp` dengan fungsi membentuk berkas PDF menggunakan *class* PDFx⁽²⁾ melalui program `p_nangpdf.prp`. Hasil proses berupa berkas PDF akan disimpan di sub folder `\pdf\`⁽³⁾.

Proses penggabungan seluruh berkas PDF pada sub folder \pdf\ dilakukan dengan menggunakan *tools* PDFtk⁽⁴⁾ untuk selanjutnya didistribusikan⁽⁵⁾.

Pada akhir proses *procedure* hapus berkas PDF dijalankan untuk mengosongkan sub folder \pdf\ sehingga siap digunakan untuk proses selanjutnya. Proses gabung berkas PDF, distribusi, dan hapus berkas merupakan bagian dari program `p_antri_petikan.prg`.

A. Pembentukan Berkas PDF DIPA Petikan

Berikut contoh *procedure* pada program `p_petikan.prg` untuk proses pembentukan berkas PDF DIPA Petikan Halaman 1A (*procedure cetak_dipa_h1a*) :

```
PARAMETERS pCetak && P:pdf T:Tayang C or empty:cetak
... //skip
PROCEDURE cetak_dipa_h1a
... //skip
IF pCetak = 'T'
  oCtk = ' Preview Noconsole'
ELSE
  oCtk = ' to Printer prompt noconsole'
ENDIF
... //skip
IF pCetak = 'P'
  nmreport = 'r_cdipa_h1a'
  namafile = 'pdf\' + pnomorsp + '_e'
  DO p_nangpdf WITH nmreport, namafile
ELSE
  Report Form r_cdipa_h1a &oCtk
ENDIF
... //skip
```

Parameter `P` pada program `p_petikan.prg` akan menjalankan program `p_nangpdf.prg` yang berfungsi untuk menghasilkan berkas PDF untuk setiap *report* DIPA Petikan.

Program `p_nangpdf.prg` memiliki dua parameter yaitu `nmreport` dan `namafile`.

Parameter `par1` (`nmreport`) menampung nama *report* (`.frx`) yang akan diproses menjadi berkas PDF, parameter `par2` (`namafile`) menampung lokasi dan nama berkas PDF yang dihasilkan. Pada *procedure* diatas *report* DIPA Petikan halaman 1A (`r_cdipa_h1a.frx`) diproses menjadi berkas PDF dengan nama berkas `pnomorsp_e.pdf` (dimana `pnomorsp` mengacu pada kode Satker) dan disimpan pada folder `pdf`.

Program `p_nangpdf.prg` berisi perintah sebagai berikut :

```
PARAMETERS prReport, prFile

PUBLIC loListener As "PdfListener" Of "Pdfx.vcx"
loListener = NewObject('PdfListener', 'PDFx.vcx')
```

```
loListener.cCodePage = "CP1252" &&CodePage
loListener.cTargetFileName = prFile+'.pdf'
loListener.QuietMode = .T.
loListener.lCanPrint = .T.
loListener.lCanEdit = .F.
loListener.lCanCopy = .T.
loListener.lCanAddNotes = .F.
loListener.lEncryptDocument = .F.
loListener.cMasterPassword = ""
loListener.cUserPassword = ""
loListener.lOpenViewer = .F.
Report Form (prReport) Object loListener
```









Parameter `prReport` merupakan nama report yang akan diproses menjadi PDF file sedangkan parameter `prFile` merupakan berkas PDF yang akan dihasilkan.

Untuk menghasilkan satu berkas PDF DIPA Petikan dengan urutan halaman DIPA sebagaimana ketentuan maka diberlakukan konversi report (parameter `prReport`) dan berkas PDF (parameter `prFile`) sebagai berikut :

No	Nama Report	DIPA Petikan	Berkas PDF
1	r_cdipa_sp	SP DIPA	pnomorsp_a.pdf
2	r_cdipa_sp_lamp	Lampiran SP DIPA (jika ada)	pnomorsp_b.pdf
3	r_cdipa_sp_lamp2	Lampiran 2 SP DIPA (jika ada)	pnomorsp_c.pdf
4	r_cdipa_sp_lamp3	Lampiran 3 SP DIPA (jika ada)	pnomorsp_d.pdf
5	r_cdipa_h1a	Halaman 1a DIPA	pnomorsp_e.pdf
6	r_cdipa_l2	Lampiran Hal 2 DIPA (jika ada)	pnomorsp_f.pdf
7	r_cdipa_h1b	Halaman 1b DIPA	pnomorsp_g.pdf
8	r_cdipa_h2	Halaman 2 DIPA	pnomorsp_h.pdf
9	r_cdipa_h2a	Lampiran Halaman 2a DIPA	pnomorsp_i.pdf
10	r_cdipa_h2b	Lampiran Halaman 2b DIPA	pnomorsp_j.pdf
11	r_cdipa_h2c	Lampiran Halaman 2c DIPA	pnomorsp_k.pdf
12	r_cdipa_h2d	Lampiran Halaman 2d DIPA	pnomorsp_l.pdf
13	r_cdipa_h3	Halaman 3 DIPA	pnomorsp_m.pdf
14	r_cdipa_h4	Halaman 4 DIPA	pnomorsp_n.pdf

Tabel1. Konversi report dan berkas PDF

Berikut contoh hasil berkas PDF pada folder `tmp` untuk satker 219056 :

Name ^	Type	Size
 219056_a	PDF File	13 KB
 219056_e	PDF File	20 KB
 219056_g	PDF File	15 KB
 219056_h	PDF File	14 KB
 219056_m	PDF File	15 KB
 219056_n	PDF File	15 KB
 libconv2	DLL File	956 KB
 pdftk	Application	5,738 KB

Gambar 1. Contoh hasil proses PDFx Satker 219056

DIPA Petikan Satker dengan kode Satker 219056 akan terdiri dari Halaman Surat Pengesahan (SP) DIPA (_a), Halaman 1A (_e), Halaman 1B (_g), Halaman 2 (_h), Halaman 3(_m), dan Halaman 4 (_n).

B. Penggabungan Berkas PDF DIPA Petikan

Penggabungan berkas PDF DIPA Petikan dilakukan oleh program `p_antri_petikan.prg` dengan perintah sebagai berikut :

```
...  
p_executeandwait("pdftk "+c_nomorsp.nomorsp+"_*.pdf cat output "+hslPDF,.f.,.t.)  
...
```

`p_executeandwait` merupakan program yang berfungsi untuk menjalankan program eksternal melalui *command prompt* di *background process* tanpa menampilkan jendela *DOS command*. Perintah diatas identik dengan perintah :

```
!pdftk "+c_nomorsp.nomorsp+"_*.pdf cat output "+hslPDF
```

`c_nomorsp.nomorsp` merupakan cursor dengan data satker yang akan di proses. `hslPDF` merupakan variabel yang menampung nama berkas PDF hasil proses penggabungan dengan format nama file sebagai berikut :

```
[tahun_anggaran]-[jenis_dokumen]-[kode_kl]-[kode_unit]-[kode_satker]-[kode_lokasi]-  
[kode_kppn]-[kode_kewenangan]-[nomor_revisi].pdf
```

untuk Satker dengan kode Satker 219056 sebagai mana contoh diatas untuk DIPA awal akan memiliki nama file : 2014-01-040-05-219056-21-061-3-00.pdf

berikut adalah bagian program `p_antri_petikan.prg` yang menunjukkan hubungan antara proses pembentukan berkas PDF dan penggabungannya menjadi satu berkas dokumen DIPA Petikan :

```
... //skip  
SELE c_nomorsp  
GO TOP  
DO WHILE !EOF()  
    SELECT c_nomorsp  
    IF sp+h1a+h1b+h2+h3+h4 > 0  
        p_petikan(pCetak) && Create PDF  
        loc0 = SYS(5)+CURDIR()  
        loc1 = loc0+'.pdf\  
        SET DEFAULT TO &loc1  
    ... //skip
```

```
hslPDF = c_nomorsp.Thang+'-'+ c_nomorsp.Kdjendok+'-'  
'+c_nomorsp.kddept+'.'+c_nomorsp.kdunit+'-'+c_nomorsp.nomorsp+'-'  
'+c_nomorsp.kdlokasi+'-'+ctkKPPN.kdkppn+'-'+ c_nomorsp.kdkwen+'-'  
'+TRANSFORM(mnorev,'@L 99')+'.pdf'  
... //skip  
p_executeandwait("pdftk "+c_nomorsp.nomorsp+"_*_.pdf cat output "+hslPDF,.f.,.t.)  
... //skip  
hapus = loc1+'*_.pdf'  
DELETE FILE &hapus  
... //skip  
SELECT c_nomorsp  
SKIP  
ENDDO  
... //skip
```

perintah `SET DEFAULT TO &loc1` berfungsi untuk memindahkan default direktori aplikasi ke sub folder `\pdf\` tempat dimana `pdftk.exe` dan berkas PDF DIPA yang akan digabungkan berada. Perintah `DELETE FILE &hapus` berfungsi untuk menghapus berkas PDF yang telah digabungkan setelah file hasil di distribusikan.

Pada akhir proses sub folder `\pdf\` hanya akan berisi file `pdftk.exe` dan `libiconv2.dll`.

Referensi

1. <http://en.wikipedia.org/wiki/Pdftk>
2. http://en.wikipedia.org/wiki/Portable_Document_Format
3. <http://weblogs.foxite.com/luisnavas>
4. <http://www.pdfabs.com/tools/pdftk-the-pdf-toolkit/>
5. PMK No. 171/PMK.02 Tahun 2013 tentang Petunjuk Penyusunan dan Pengesahan Daftar Isian Pelaksanaan Anggaran

Biografi Penulis

Didik Setiawan

Pranata Komputer Kementerian Keuangan RI.


```
Lparameters tcCommand, tlVisible, tlWait
Local lcStart, lcProcess, lcCmd, lnHandle, llSuccess
lcCmd = tcCommand
llSuccess = .F.

*-- example of use:
*-- ExecuteAndWait("notepad.exe", .t., .t.)

*_cliptext = tcCommand
Declare Integer CreateProcess In "kernel32.dll" ;
    as waCreateProcess ;
    integer    lpApplicationName, ;
    string     lpCommandLine, ;
    integer    lpProcessAttributes, ;
    integer    lpThreadAttributes, ;
    integer    bInheritHandles, ;
    integer    dwCreationFlags, ;
    integer    lpEnvironment, ;
    integer    lpCurrentDirectory, ;
    string     @lpStartupInfo, ;
    string     @lpProcessInformation

Declare Integer WaitForSingleObject In kernel32.Dll ;
    as waWaitForSingleObject ;
    integer hHandle, Integer dwMilliseconds

Declare Integer CloseHandle In kernel32.Dll ;
    as waCloseHandle ;
    integer hObject
    BinToC( 68, '4rs' )
*lcStart = FLL_LONGTOCHAR(68) + Replicate(Chr(0), 64)
lcStart = BinToC( 68, '4rs' ) + Replicate(Chr(0), 64)
lcProcess = Replicate(Chr(0), 16)

lcCmd = tcCommand + Chr(0)
*-- added 0x08000000 (CREATE_NO_WINDOW)
If tlVisible Then
    = waCreateProcess(0, lcCmd, 0, 0, 1, 0 + 32, 0, 0, @lcStart, @lcProcess)
Else
    = waCreateProcess(0, lcCmd, 0, 0, 1, 0x08000000 + 32, 0, 0, @lcStart, @lcProcess)
Endif

*lnHandle = FLL_CHARTOULONG(Substr(lcProcess, 1, 4))
lnHandle = CToBin(Substr(lcProcess, 1, 4), '4rs')

*-- waiting for finish...
If tlWait Then
    = waWaitForSingleObject(lnHandle, -1)
Endif

*-- close handle
waCloseHandle(lnHandle)

*-- cleaning up
Clear Dlls waCreateProcess
Clear Dlls waWaitForSingleObject
Clear Dlls waCloseHandle

RETURN
```