The Unicode® Consortium

Unicode Implementers' Workshop 6

Santa Clara, California
September 8–9, 1994

Creating Fonts for the Unicode Kanji Set: Problems & Solutions

by

Ken Lunde
Adobe Systems, Inc.

Creating Fonts for the Unicode Kanji Set: Problems and Solutions

Dr. Ken Lunde

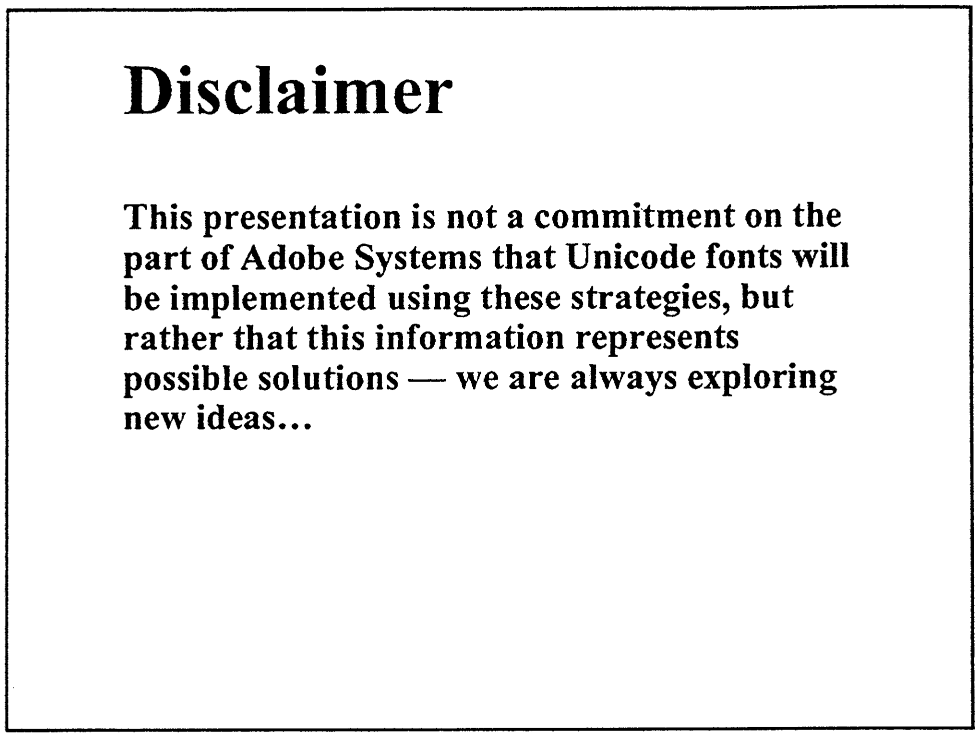
Project Manager for CJK Font Production

Adobe Systems Incorporated

This presentation provides developers with information on how the approximately 21,000 characters in the Unicode kanji set can be handled in font development.

Dr. Lunde has been working at Adobe Systems since 1991, and is currently the Project Manager for CJK Font Production. He holds BA (1987), MA (1988), and PhD (1994) degrees in Linguistics from The University of Wisconsin-Madison. He has authored numerous papers and articles on the subject of Japanese character sets and encodings, and has most recently written a book entitled *Understanding Japanese Information Processing* (O'Reilly & Associates, Inc., 1993, ISBN 1-56592-043-0). His e-mail address is `lunde@mv.us.adobe.com`.

Adobe Systems is a leader in font-related technologies, and thus has a vested interest in Unicode.

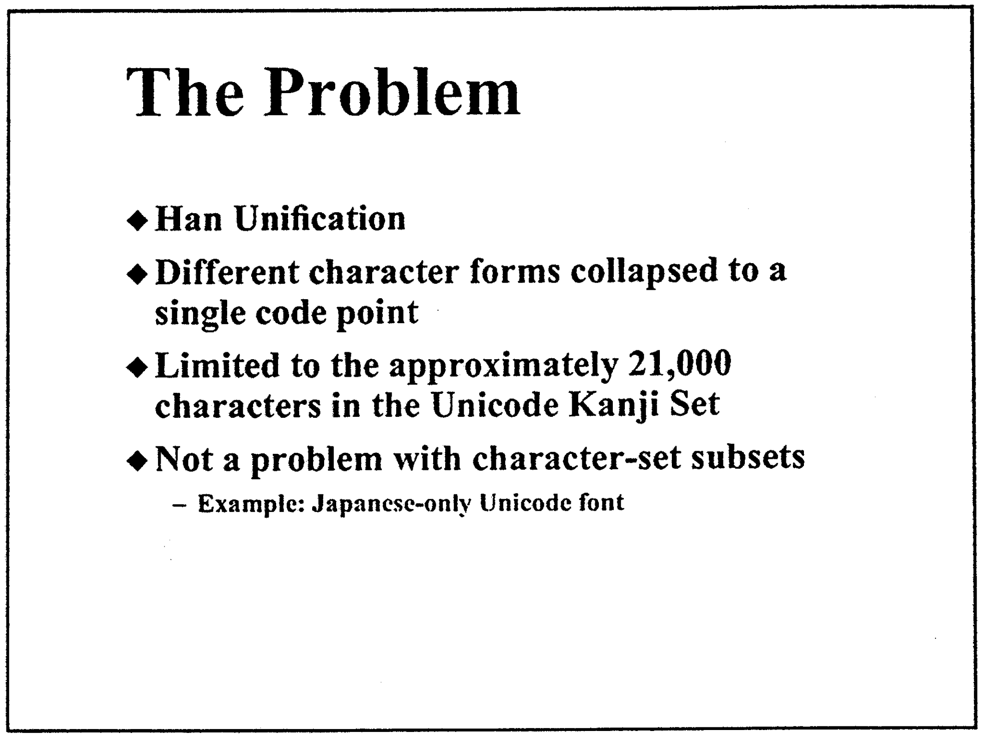


Disclaimer

This presentation is not a commitment on the part of Adobe Systems that Unicode fonts will be implemented using these strategies, but rather that this information represents possible solutions — we are always exploring new ideas...

Don't get too worried about this statement. This is to simply warn you that the material presented here does not necessarily represent how Adobe Systems will finally decide to implement Unicode-based fonts.

This presentation should indicate that Adobe Systems is working, just like many other developers, on solutions for implementing Unicode-based products.



The Problem

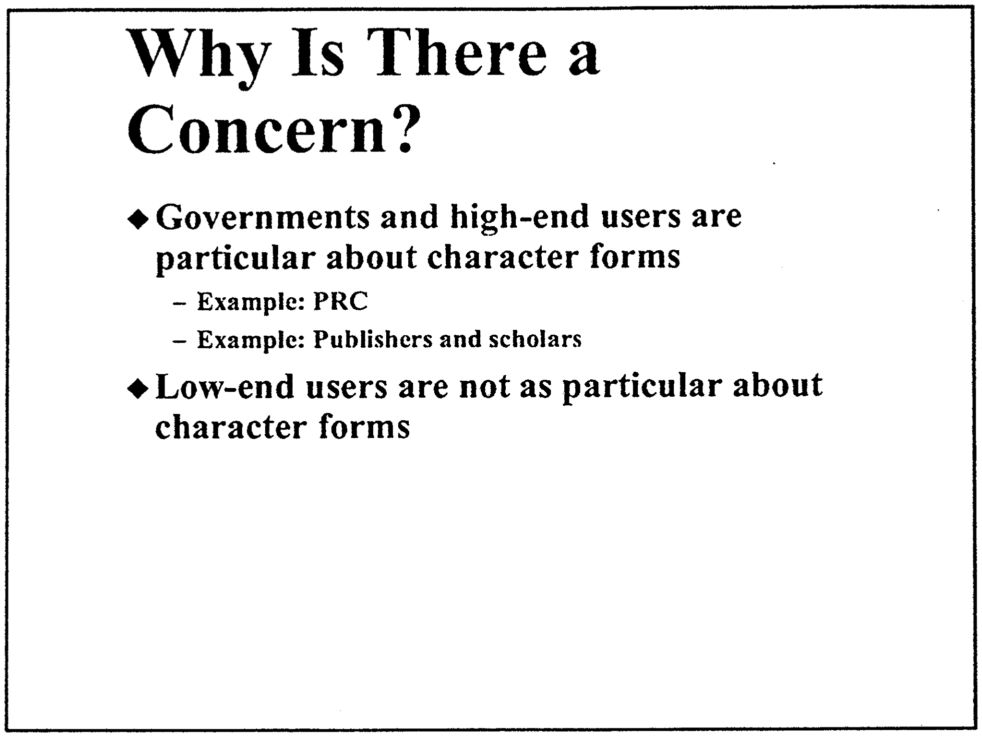
- ◆ **Han Unification**
- ◆ **Different character forms collapsed to a single code point**
- ◆ **Limited to the approximately 21,000 characters in the Unicode Kanji Set**
- ◆ **Not a problem with character-set subsets**
  - **Example: Japanese-only Unicode font**

The creation of the approximately 21,000 Chinese characters (kanji) in the Unicode character set came from the unification of over 120,000 kanji from dozens of Asian character set standards. This effort is known as "Han Unification."

Many kanji in these character set standards share both character form and meaning. However, slight character form differences were unified into a single code point under the process of Han Unification. Examples of such characters will be provided later in this presentation.

The focus of this presentation is limited to this set of approximately 21,000 kanji in the Unicode kanji set. These characters represent the major focus of *The Unicode Standard: Worldwide Character Encoding*, Version 1.0, Volume 2 (Addison-Wesley, 1992, ISBN 0-201-60845-6).

In many cases, developers may wish to target their product to a specific market, such as Japan. In this case, only those characters within Unicode that are used for Japanese need to be addressed. The results of Han Unification become a non-issue in such cases because the character forms can be tailored to the specific market. However, other issues come up, and must be addressed.



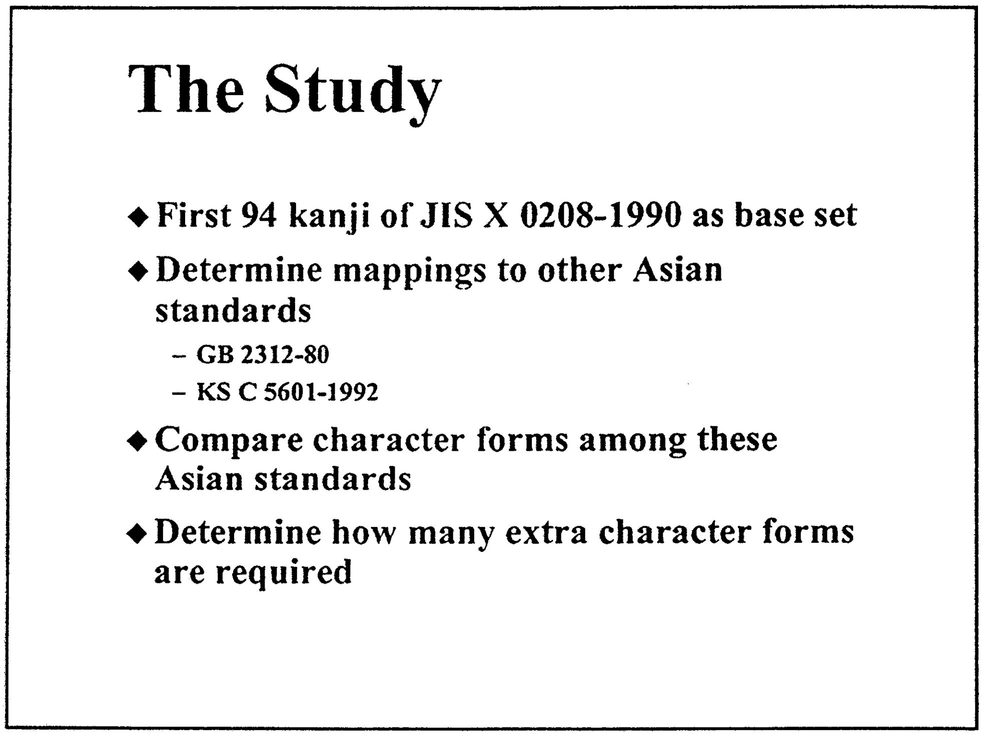
Why Is There a Concern?

- ◆ **Governments and high-end users are particular about character forms**
  - **– Example: PRC**
  - **– Example: Publishers and scholars**
- ◆ **Low-end users are not as particular about character forms**

There are several issues raised when discussing kanji character forms. There are those who claim that character form differences do not matter, citing evidence that the exact character form has no relation to meaning.

It is true that character form is a non-issue for many users, but some governments, most publishers, and scholars have very strong opinions on character forms. If a product does not conform to certain character form standards, it will have a difficult time gaining acceptance.

This means that character form can be an important issue when designing a Unicode-based font product. It can become even more difficult when a single font must satisfy Chinese, Japanese, and Korean users.



The Study

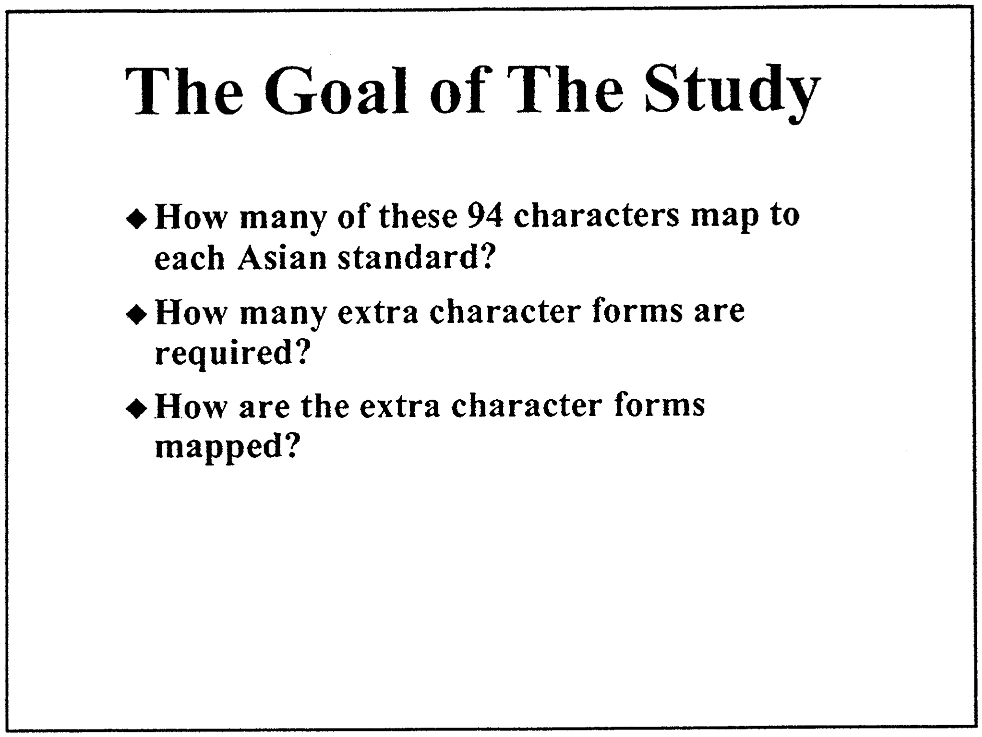
- ◆ First 94 kanji of JIS X 0208-1990 as base set
- ◆ Determine mappings to other Asian standards
  - – GB 2312-80
  - – KS C 5601-1992
- ◆ Compare character forms among these Asian standards
- ◆ Determine how many extra character forms are required

The purpose of this study was to determine to what extent differing character forms were collapsed under Han Unification. I decided to use the first 94 kanji of JIS X 0208-1990 as the base set for this study.

The first task is to determine how these 94 characters map to other Asian standards. For the purpose of this study, only GB 2312-80 (PRC) and KS C 5601-1992 (Korea) were chosen.

The next task is to compare the character forms of the characters that do map to other Asian standards. Consulting the standards manuals and other references is a crucial part of this task.

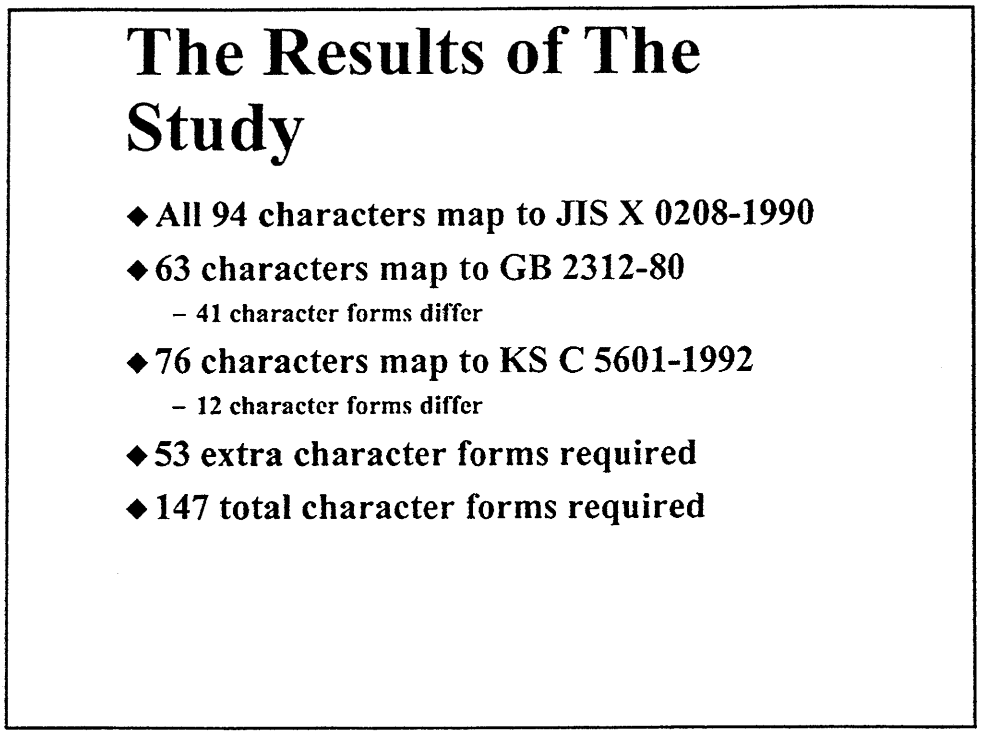
The differences in character form will then determine just how many additional character forms are required in order to fully represent these 94 characters in Japanese, Chinese, and Korean.



The Goal of The Study

- How many of these 94 characters map to each Asian standard?
- How many extra character forms are required?
- How are the extra character forms mapped?

This study has several goals, enumerated above. Although this study deals with a small set of characters, the results can be extended to the entire set of Unicode kanji.



The Results of The Study

- All 94 characters map to JIS X 0208-1990
- 63 characters map to GB 2312-80
  - 41 character forms differ
- 76 characters map to KS C 5601-1992
  - 12 character forms differ
- 53 extra character forms required
- 147 total character forms required

The results of this study showed that all 94 of these characters map to JIS X 0208-1990 (obviously), 63 map to GB 2312-80, and 76 map to KS C 5601-1992.

More interesting is that of these 63 that map to GB 2312-80, 41 have different character forms than Japanese and Korean. Likewise, of the 76 that map to KS C 5601-1992, 12 have different character forms than Japanese and Chinese.

This means that in addition to the base 94 character forms used for Japanese, 53 additional character forms are required to express these 94 characters in Chinese and Korean. This totals to 147 characters.

If we extend to the complete Unicode kanji set this ratio of base characters to additional character forms, we find that up to 10,000 additional character forms may be required.

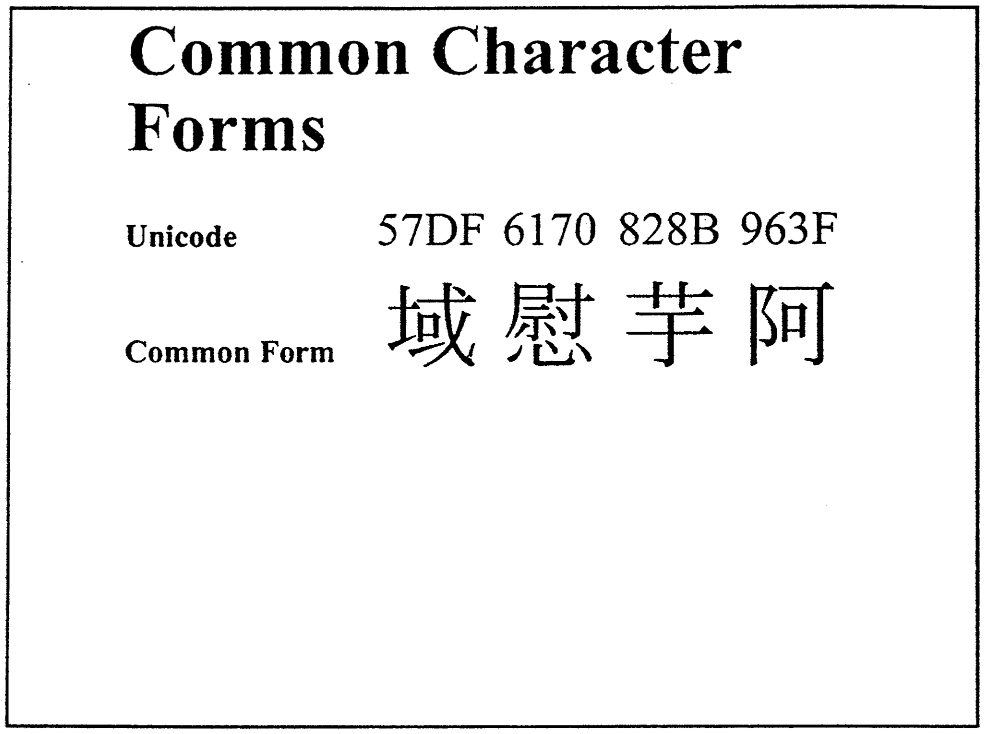
The question now is how a single font can handle these multiple character forms assigned to a single code point, but first let's examine these character forms a bit more closely...

The following three pages are the results in printed form, showing those characters that do and do not map to Chinese and Korean. An asterisk means that the character does not map, an encoded value (in row-cell format) means that the character does map, and an encoded value plus character forms means that that mapping does occur, but requires a different character form.

Unicode	JIS X 0208-1990		GB 2312-80		KS C 5601-1992
4E00	一	1676		5027	7673
4E95	井	1670		3014	7944
4E9C	亜	1601		*	*
4EA5	亥	1671	亥	2605	9004
4EE5	以	1642	以	5052	7604
4F0A	伊	1643		5033	7605
4F4D	位	1644	位	4627	7440
4F9D	依	1645	依	5032	7578
5049	偉	1646		*	7441
5141	允	1684	允	5242	7535
533B	医	1669	医	5029	*
5370	印	1685	印	5101	7652
54BD	咽	1686		4942	7654
54C0	哀	1605	哀	1607	6878
54E1	員	1687		*	7412
5516	唖	1602		*	*
56E0	因	1688	因	5082	7655
56F2	囲	1647		*	*
5727	圧	1621		*	*
57DF	域	1672		5182	7020
58F1	壱	1677		*	*
5937	夷	1648	夷	5036	7608
59D0	姐	1625	姐	2967	7827
59D4	委	1649	委	4615	7445
59F6	姶	1608		*	*
59FB	姻	1689	姻	5086	7656
5A01	威	1650	威	4594	7446
5A03	娃	1603	娃	4562	7263
5B89	安	1634	安	1618	6844
5B9B	宛	1624	宛	4580	7240
5C09	尉	1651		4630	7447

5EB5	庵	1635	庵	6654		6861
5F15	引	1690	引	5093		7658
60AA	悪	1613		*		*
60DF	惟	1652	惟	4609		7478
610F	意	1653	意	5066		7582
611B	愛	1606		*		6881
6170	慰	1654		4631		7448
6216	或	1631		2782		9168
6271	扱	1623		*	扱	4866
6309	按	1636	按	1620		6846
6328	挨	1607	挨	1604		*
63E1	握	1614	握	4653		6836
65A1	斡	1622		4651		6854
65ED	旭	1616		4881		7379
6613	易	1655		5055		7022
6697	暗	1637	暗	1621	暗	6862
674F	杏	1641		4851		9026
6848	案	1638	案	1624		6848
6893	梓	1620	梓	7287		7809
6905	椅	1656		5046		7585
6DEB	淫	1692		5089	淫	7566
6E25	渥	1615	渥	6855		6838
6EA2	溢	1678	溢	5071	溢	7678
70BA	為	1657		*		*
754F	畏	1658	畏	4623		7270
7570	異	1659		*		7622
78EF	磯	1675		*	磯	4920
79FB	移	1660		5038		7625
7A32	稲	1680		*		*
7A50	穐	1612		*		*
7C9F	粟	1632		4358		6556
7D62	絢	1628		*		9066

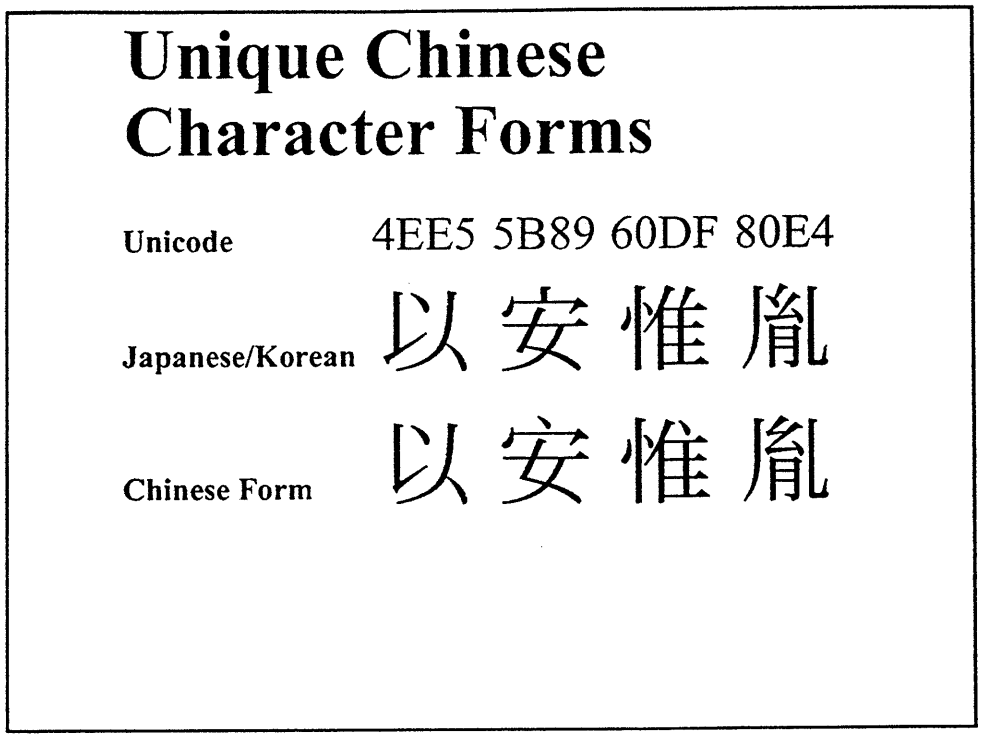
7DAD	維	1661		*		7511
7DBE	綾	1629		*		5551
7DEF	緯	1662		*		7453
80B2	育	1673	育	5193		7532
80C3	胃	1663		4624		7454
80E4	胤	1693	胤	5623		7542
828B	芋	1682		5183		7367
82A6	芦	1618	芦	3411	芦	9156
831C	茜	1611		6071		*
8328	茨	1681	茨	2036	茨	7728
840E	萎	1664	萎	4614		7455
8466	葦	1617		*		7456
8475	葵	1610	葵	3191		4813
852D	蔭	1694		*		7567
867B	虻	1626	虻	8221		*
8863	衣	1665	衣	5034		7593
88B7	袷	1633	袷	8142		*
8B02	謂	1666		*		7461
9022	逢	1609		2374	逢	6081
9038	逸	1679	逸	5061	逸	7679
9055	違	1667		*	違	7462
907A	遺	1668		*	遺	7522
90C1	郁	1674		5184		7384
95C7	闇	1639		*		6865
963F	阿	1604		1602		6825
978D	鞍	1640	鞍	1616		6851
98F2	飲	1691		*		*
98F4	飴	1627		*	飴	7639
9B8E	鮎	1630		*		7938
9BF5	鯵	1619		*		*
9C2F	鰯	1683		*		*



Common Character Forms

Unicode	57DF	6170	828B	963F
Common Form	域	慰	芋	阿

Many characters, regardless of their type design, share a common form in Chinese, Japanese, and Korean. This slide provides four examples from the first 94 characters in JIS X 0208-1990.

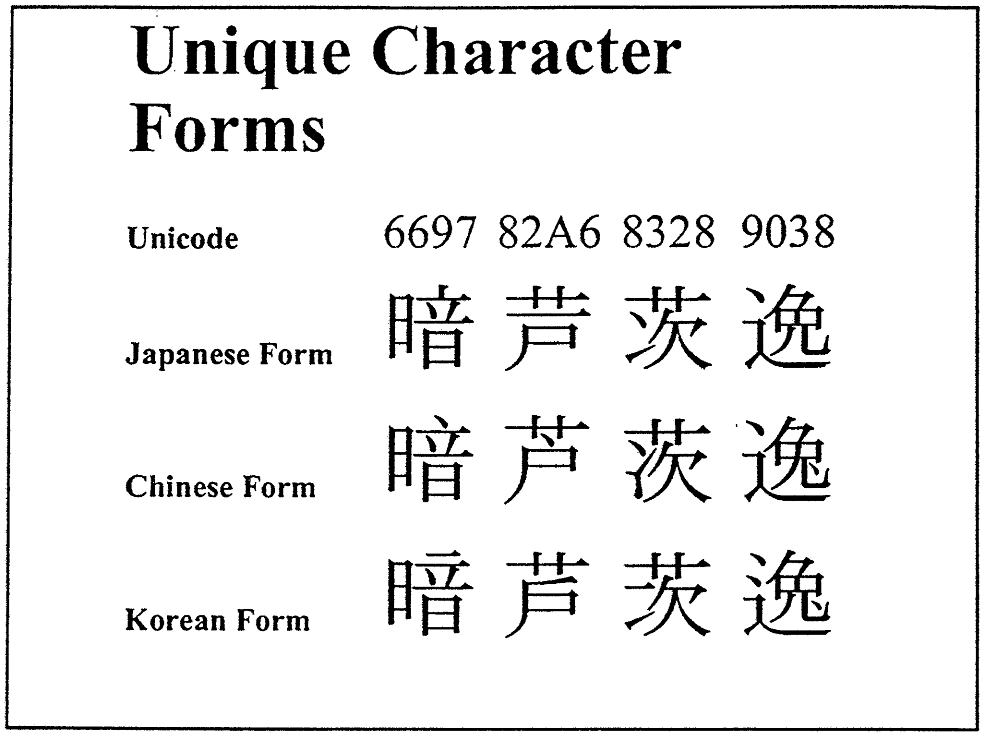


Unique Chinese Character Forms

Unicode	4EE5	5B89	60DF	80E4
Japanese/Korean	以	安	惟	胤
Chinese Form	以	安	惟	胤

Some characters have a form unique to Chinese. This slide provides four examples taken from the first 94 characters in JIS X 0208-1990. The Chinese forms are those in GB 2312-80. Some characters illustrate more than one difference.

GB 6345.1-86 is the document that specifies character forms for the characters in GB 2312-80. The Chinese form actually more closely represents how characters are written with a brush.

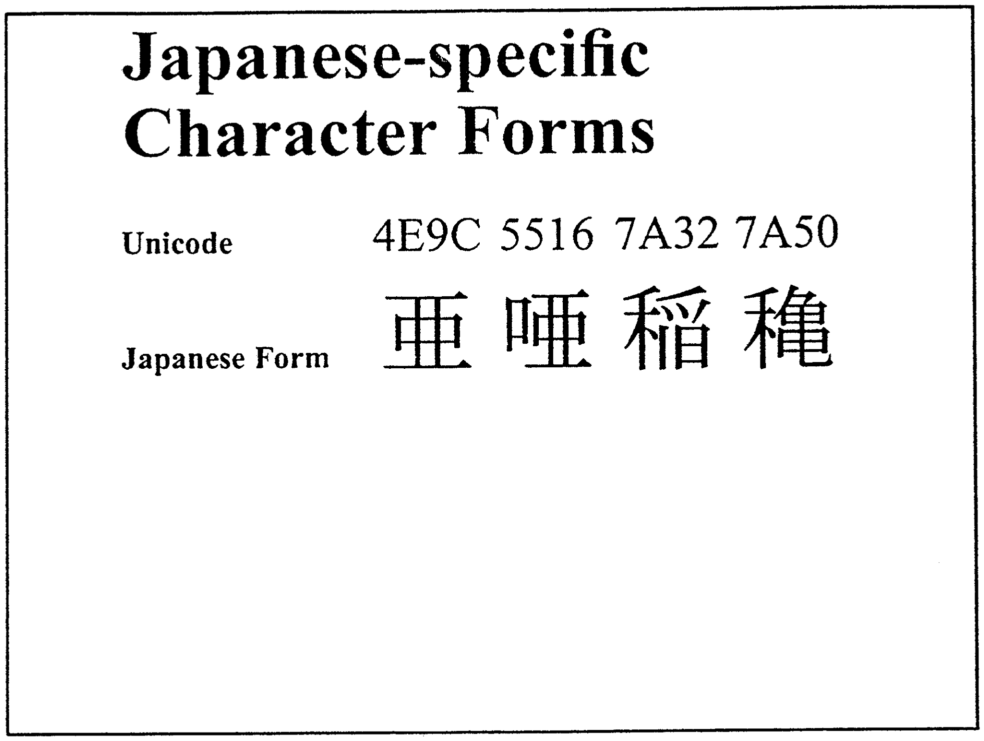


Unique Character Forms

Unicode	6697	82A6	8328	9038
Japanese Form	暗	芦	茨	逸
Chinese Form	暗	芦	茨	逸
Korean Form	暗	芦	茨	逸

Some characters have a form unique to Chinese, Japanese, and Korean. This slide provides four examples taken from the first 94 characters in JIS X 0208-1990. The Chinese forms are those in GB 2312-80, and the Korean forms are those in KS C 5601-1992. Some characters illustrate more than one difference.

The difference between the Japanese and Korean character forms is typically a case of simplified versus traditional form. In fact, many of the Korean character forms are identical to forms found in the 1978 version of JIS X 0208 (JIS C 6226-1978).

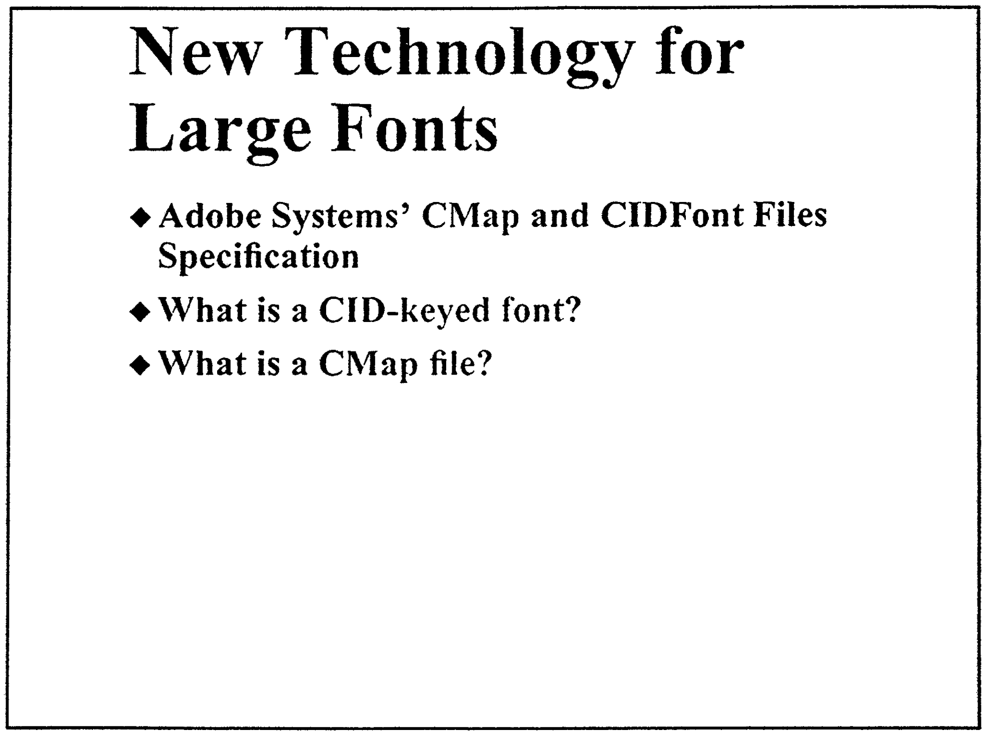


Japanese-specific Character Forms

Unicode	4E9C	5516	7A32	7A50
Japanese Form	亜	唖	稲	穐

Some characters have a form unique to Japanese, and are unused in Chinese and Korean. This slide provides four examples taken from the first 94 characters in JIS X 0208-1990.

Note, however, that variants of these characters are used in Chinese and Korean. The character forms illustrated here are simplified forms that are unique to Japan—the unsimplified forms, which are used in Chinese and Korean, exist at a different Unicode code point.



New Technology for Large Fonts

- Adobe Systems' CMap and CIDFont Files Specification
- What is a CID-keyed font?
- What is a CMap file?

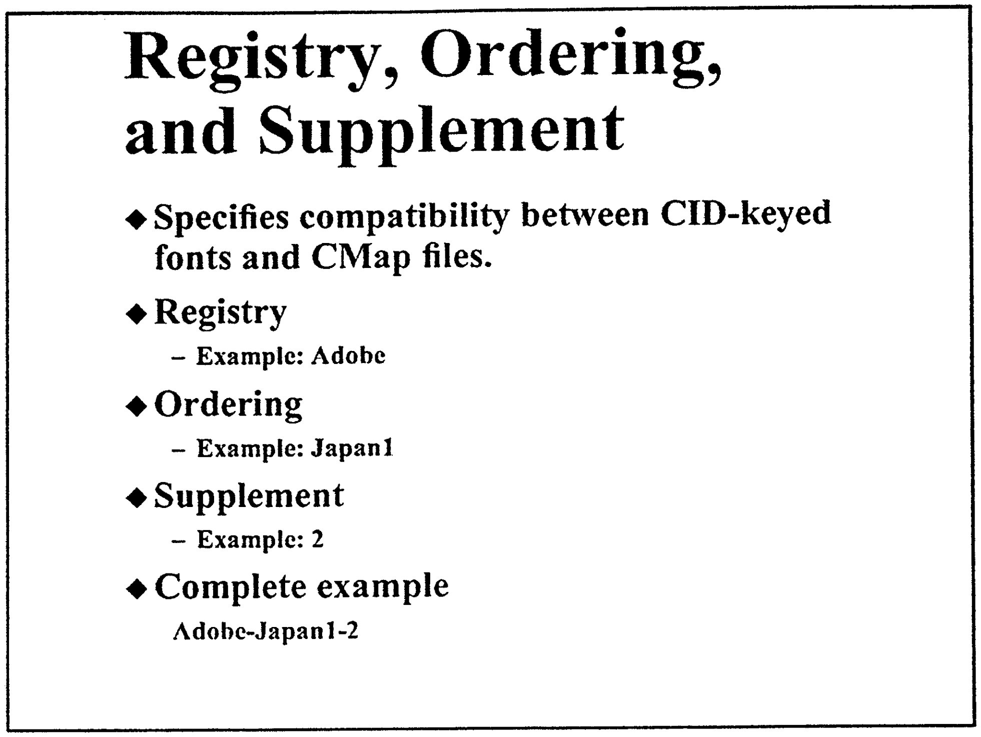
Adobe Systems Incorporated has recently developed the CMap and CIDFont Files Specification. This technology is most useful in representing fonts with large character sets, such as those for Chinese, Japanese, or Korean.

One component is called the CID-keyed font. CID stands for "Character ID," and is simply a unique ID assigned to each character in the font. But a CID is not an encoding, but just an enumerated value with which a character can be accessed. The CID-keyed font can be thought of as a "character bucket."

The other component is one or more CMap files. CMap stands for "Character Map," and provides the encoding-to-CID correspondences.

A complete font is composed of two names: a CID-keyed font name, such as "Ming-Medium"; and a CMap name, such as "90ms-RKSJ-H." These are then attached with a hyphen or double-hyphen (recommended), making "Ming-Medium-90ms-RKSJ-H" or "Ming-Medium--90ms-RKSJ-H." This is the PostScript font name that is subsequently sent to the printer or other output device for imaging. The CID technology then parses the font name, and produces a Type 0 font in virtual memory that supports the desired character set and encoding.

The CMap "90ms-RKSJ-H" has several components, delimited by hyphens, that are embedded in its name. "90ms" stands for the Microsoft Windows 3.1J character set. "RKSJ" stands for Shift-JIS encoding. "H" stands for a horizontal writing mode (as opposed to a vertical writing mode).



Registry, Ordering, and Supplement

- ◆ Specifies compatibility between CID-keyed fonts and CMap files.
- ◆ Registry
  - Example: Adobe
- ◆ Ordering
  - Example: Japan1
- ◆ Supplement
  - Example: 2
- ◆ Complete example

  Adobe-Japan1-2

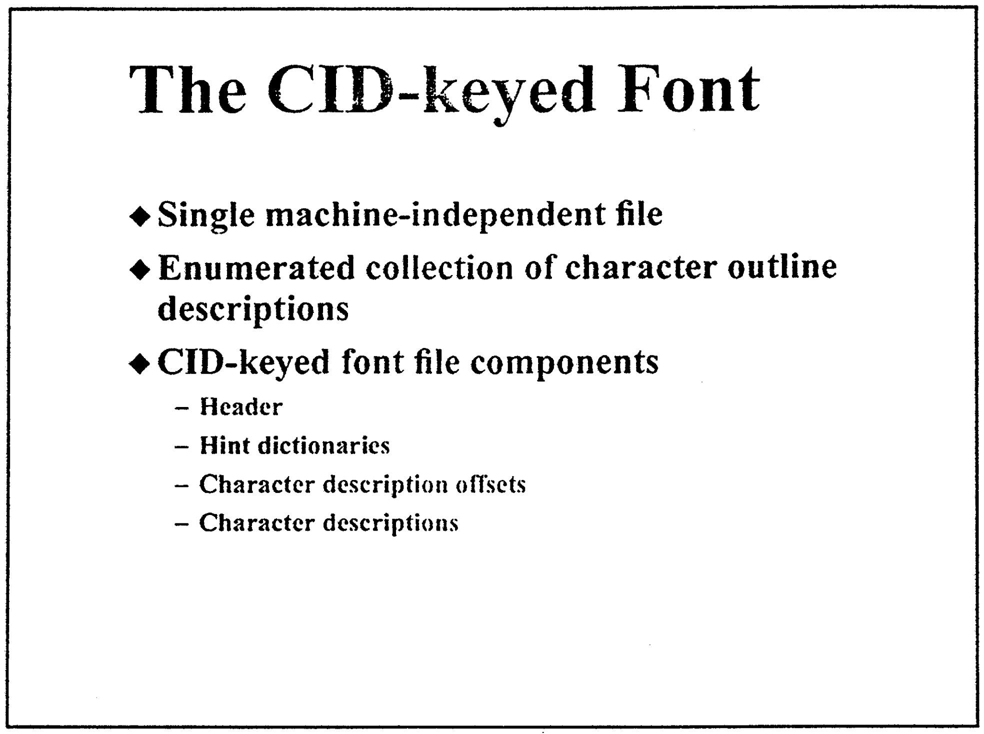
This new technology uses three pieces of information to link CID-keyed fonts with their corresponding CMap files.

The Registry is the originator of the character collection. In the case of those developed by Adobe Systems, the Registry field is "Adobe."

The Ordering is the actual name given to the character collection, and typically adheres to a specific character set standard and its variations. Adobe Systems calls its standard Japanese character collection "Japan1," and supports the JIS X 0208-1990 standard and its variations (such as earlier versions and corporate character sets based on it).

The Supplement indicates the version of the character collection. The first Supplement is always "0," and is the base collection of characters. For example, Adobe-Japan1-0 enumerates 8,284 characters, Adobe-Japan1-1 enumerates 8,359 characters (75 are in Supplement 1), and Adobe-Japan1-2 enumerates 8,720 characters (361 are in Supplement 2).

Why supplements? National and corporate character sets often change, usually by adding characters. Supplement 0 is based on JIS X 0208-1983, and supports the NEC, Fujitsu, and Apple variations. Supplement 1 is based on JIS X 0208-1990, and adds support for Apple's KanjiTalk 7.1 character set. Supplement 2 adds support for the Microsoft Windows 3.1J character set.



The CID-keyed Font

- ◆ Single machine-independent file
- ◆ Enumerated collection of character outline descriptions
- ◆ CID-keyed font file components
  - Header
  - Hint dictionaries
  - Character description offsets
  - Character descriptions

A CID-keyed font is simply an enumerated collection of character outline descriptions along with other font-related information.

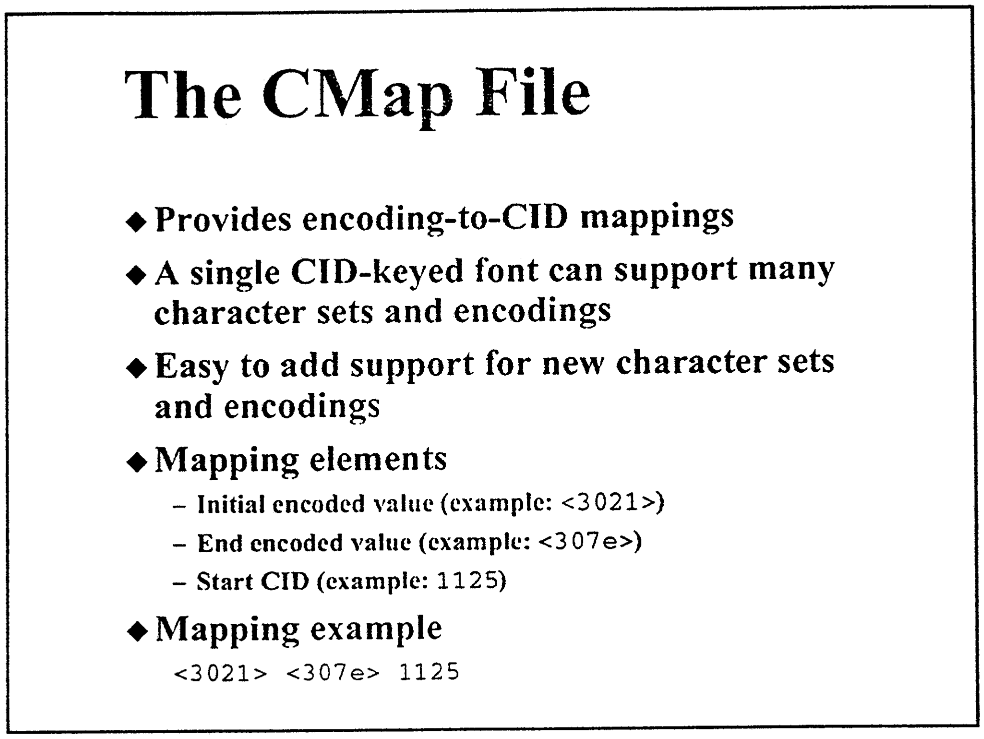
The header contains information about the font, such as version number, registry, ordering, supplement, and number of characters.

The hint dictionary contains information that improves output quality at small point sizes and at low resolutions.

The character description offsets simply tell the machinery where the individual character description is located within the CID-keyed font file.

The character descriptions are the actual code that describes the shape of the character as a series of lines, arcs, and curves.

The same CID-keyed font, without any modifications, works on a variety of output devices, such as printers and image-setters as well as with host-based solutions running on a variety of machines, such as ATM (Adobe Type Manager), CPSI (Configurable PostScript Interpreter), and DPS (Display PostScript).



The CMap File

- ◆ **Provides encoding-to-CID mappings**
- ◆ **A single CID-keyed font can support many character sets and encodings**
- ◆ **Easy to add support for new character sets and encodings**
- ◆ **Mapping elements**
  - **Initial encoded value (example: `<3021>`)**
  - **End encoded value (example: `<307e>`)**
  - **Start CID (example: `1125`)**
- ◆ **Mapping example**

```
<3021> <307e> 1125
```

The CMap file provides the vital link between encoded character values, produced by an application, and CIDs.

Through the use of multiple CMap files, a single CID-keyed font can support a large number of character set and encoding combinations. This also means that adding support of other character sets and encodings, such as Unicode, can be accomplished by issuing a new CMap, rather than issuing a new version of the CID-keyed font.

A CMap file is a simple textfile, and anyone with a good foundation in character sets and encodings can craft these.

Besides the header of the CMap file, which provides compatibility and code space information, there are the actual entries that define the encoding-to-CID mappings. These consist of three components.

The first component is the initial encoded value for a (possible) range of contiguous encodings. The example, `<3021>`, represents the first character in the first row of 94 kanji in JIS X 0208-1990.

The second component is the end encoded value for the range of contiguous encodings. The example, `<307e>`, represents the last character in the first row of 94 kanji in JIS X 0208-1990. Thus, "`<3021> <307e>`" represents a range of 94 contiguous encoded values.

The third component is the starting CID for the encoded range. The example, `1125`, means that 0x3021 is assigned CID 1125, 0x3022 is assigned CID 1126, and so on until 0x307e, the end encoded value, is reached.

Encoding-to-CID Mapping Issues

- **Contiguous encoding ranges and contiguous CIDs result in increased efficiency**
- **Dense mapping examples**

`<3021> <307e> 1125`	**(94 characters)**
`<4e00> <4eff> 2057`	**(256 characters)**

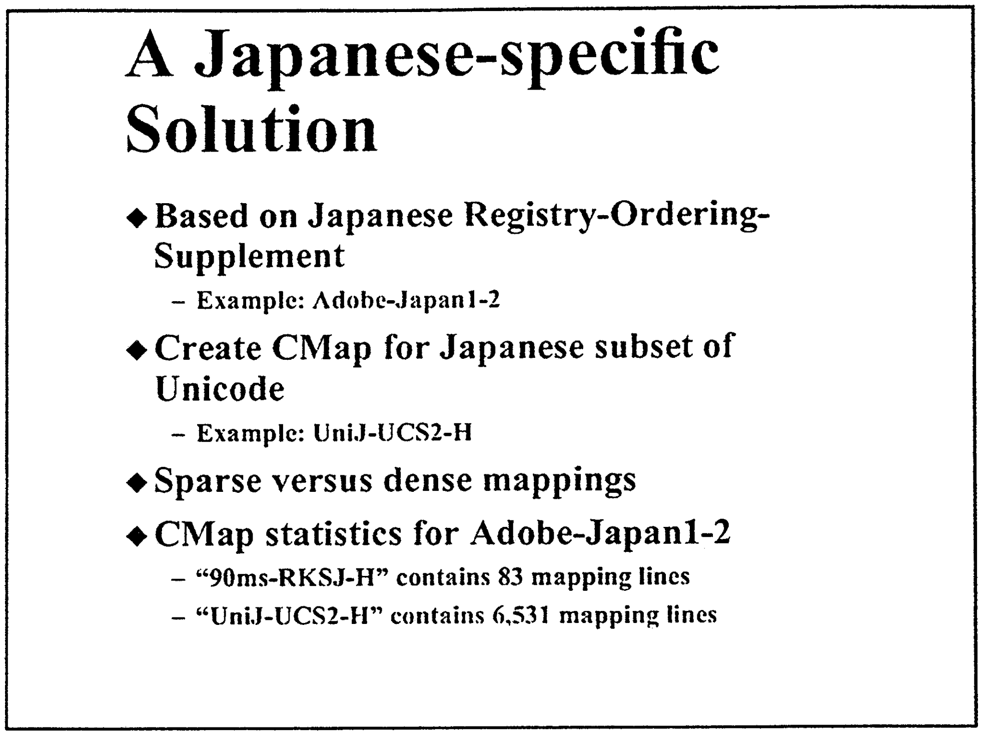
- **Sparse mapping examples**

`<4e00> <4e00> 2057`	**(1 character)**
`<4e01> <4e01> 2172`	**(1 character)**
`<4e02> <4e02> 3971`	**(1 character)**
`<4e03> <4e05> 1054`	**(3 characters)**

One always hopes for both contiguous code points *and* contiguous CIDs. In the case of Adobe-Japan1-2, the ordering of the kanji are such that this is typically the case, at least for the native Japanese encodings, JIS, Shift-JIS, and EUC. Such contiguity results in what we call "dense" mapping, which leads to a smaller CMap file, thus increased efficiency.

Dense mappings are those that cover a wide range of encoded values, usually an entire first byte value, such as "`<3021> <307e>`" for JIS encoding or "`<4e00> <4eff>`" for UCS-2. Those cover 94 and 256 characters, respectively.

Sparse mappings are those that cover small encoded ranges, sometimes consisting of only one character, such as "`<4e00> <4e00>`." These usually come up when dealing with character sets and encodings with a different ordering than what is in the CID-keyed font. This is the case when applying Unicode ordering to a Japanese CID-keyed font.



A Japanese-specific Solution

- Based on Japanese Registry-Ordering-Supplement
  - Example: Adobe-Japan1-2
- Create CMap for Japanese subset of Unicode
  - Example: UniJ-UCS2-H
- Sparse versus dense mappings
- CMap statistics for Adobe-Japan1-2
  - "90ms-RKSJ-H" contains 83 mapping lines
  - "UniJ-UCS2-H" contains 6,531 mapping lines

CID-keyed font products that are targeted at the Japanese market can support both native Japanese encoding methods (for example, JIS, Shift-JIS, and EUC) and Unicode—all with separate CMap files.

The most current version of Adobe Systems' Japanese character collection is Registry Adobe, Ordering Japan1, and Supplement 2 (Adobe-Japan1-2). This collection of characters is sufficient to support a wide variety of character sets and encodings.

I recently created a Unicode (UCS-2 encoding) CMap that works with Adobe-Japan1-2 CID-keyed fonts.

Adobe-Japan1-2 has its characters ordering in such a way that native Japanese encodings produce dense mappings in the CMap file. However, the ordering of characters within Unicode, especially the kanji, produces sparse mappings in the CMap file.

For example, the "90ms-RKSJ-H" CMap, which supports the Microsoft Windows 3.1J character set, contains 83 mapping entries. In contrast, the "UniJ-UCS2-H" CMap, which currently supports the identical character set but UCS-2 encoding, contains 6,531 mapping entries.

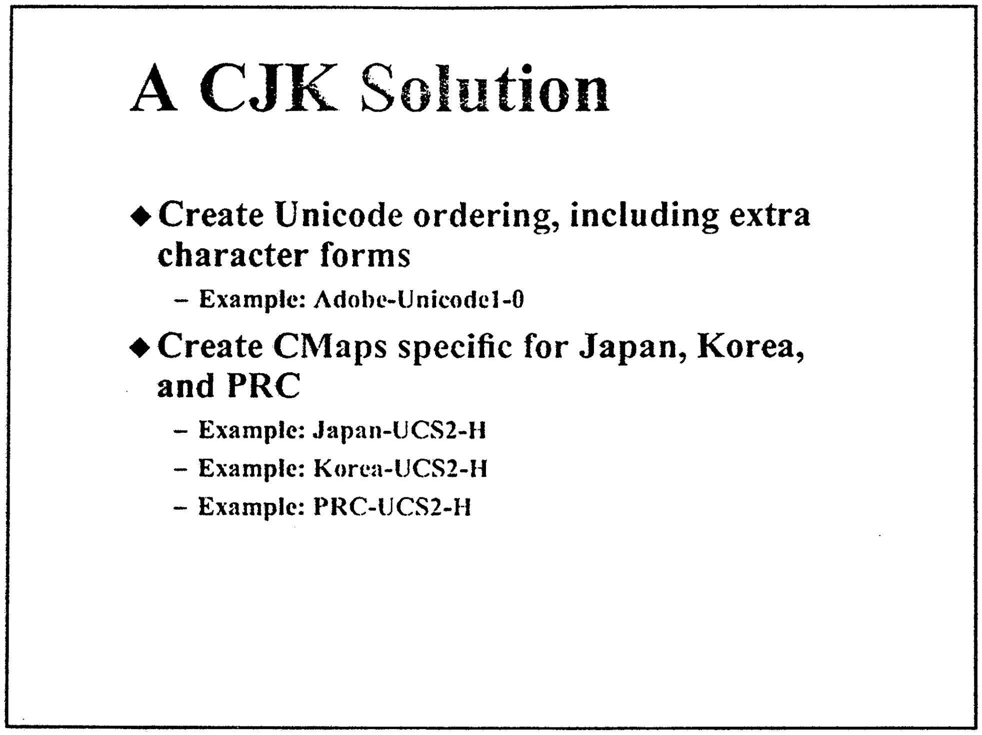
The next few pages represent the 90ms-RKSJ-H CMap and a partial UniJ-UCS2-H CMap. Note that these CMap files, as provided here, contain only kanji mappings. These are included for reference purposes only, and do not represent the final forms.

```
%!PS-Adobe-3.0 Resource-CMap
%%DocumentNeededResources: procset CIDInit
%%IncludeResource: procset CIDInit
%%BeginResource: CMap (90ms-RKSJ-H)
%%Title: (90ms-RKSJ-H Adobe Japan1 2)
%%Version: 1

/CIDInit /ProcSet findresource begin

12 dict begin

begincmap

/CIDSystemInfo 3 dict dup begin
  /Registry (Adobe) def
  /Ordering (Japan1) def
  /Supplement 2 def
end def

/CMapName /90ms-RKSJ-H def
/CMapVersion 1 def
/CMapType 1 def

/WMode 0 def

4 begincodespacerange
  <00>   <80>
  <8140> <9FFC>
  <A0>   <DF>
  <E040> <FCFC>
endcodespacerange

83 begincidrange
<889f> <88fc> 1125
<8940> <897e> 1219
<8980> <89fc> 1282
<8a40> <8a7e> 1407
<8a80> <8afc> 1470
<8b40> <8b7e> 1595
<8b80> <8bfc> 1658
<8c40> <8c7e> 1783
<8c80> <8cfc> 1846
<8d40> <8d7e> 1971
<8d80> <8dfc> 2034
<8e40> <8e7e> 2159
<8e80> <8efc> 2222
<8f40> <8f7e> 2347
<8f80> <8ffc> 2410
<9040> <907e> 2535
<9080> <90fc> 2598
<9140> <917e> 2723
<9180> <91fc> 2786
<9240> <927e> 2911
<9280> <92fc> 2974
<9340> <937e> 3099
<9380> <93fc> 3162
<9440> <947e> 3287
<9480> <94fc> 3350
<9540> <957e> 3475
<9580> <95fc> 3538
<9640> <967e> 3663
<9680> <96fc> 3726
<9740> <977e> 3851
<9780> <97fc> 3914
<9840> <9872> 4039
<989f> <98fc> 4090
<9940> <997e> 4184
<9980> <99fc> 4247
<9a40> <9a7e> 4372
<9a80> <9afc> 4435
<9b40> <9b7e> 4560
<9b80> <9bfc> 4623
<9c40> <9c7e> 4748
<9c80> <9cfc> 4811
<9d40> <9d7e> 4936
<9d80> <9dfc> 4999
<9e40> <9e7e> 5124
<9e80> <9efc> 5187
<9f40> <9f7e> 5312
<9f80> <9ffc> 5375
<e040> <e07e> 5500
<e080> <e0fc> 5563
<e140> <e17e> 5688
<e180> <e1fc> 5751
<e240> <e27e> 5876
<e280> <e2fc> 5939
<e340> <e37e> 6064
<e380> <e3fc> 6127
<e440> <e47e> 6252
<e480> <e4fc> 6315
<e540> <e57e> 6440
<e580> <e5fc> 6503
<e640> <e67e> 6628
<e680> <e6fc> 6691
<e740> <e77e> 6816
<e780> <e7fc> 6879
<e840> <e87e> 7004
<e880> <e8fc> 7067
<e940> <e97e> 7192
<e980> <e9fc> 7255
<ea40> <ea7e> 7380
<ea80> <eaa2> 7443
<eaa3> <eaa4> 8284
<ed40> <ed7e> 8359
<ed80> <edb3> 8422
<edb4> <edb4> 1993
<edb5> <edfc> 8474
<ee40> <ee7e> 8546
<ee80> <eeec> 8609
<fa5c> <fa7e> 8359
<fa80> <facf> 8394
<fad0> <fad0> 1993
<fad1> <fafc> 8474
<fb40> <fb7e> 8518
<fb80> <fbfc> 8581
<fc40> <fc4b> 8706
endcidrange
endcmap
CMapName currentdict /CMap defineresource pop
end
end

%%EndResource
%%EOF
```

```
%!PS-Adobe-3.0 Resource-CMap
%%DocumentNeededResources: procset CIDInit
%%IncludeResource: procset CIDInit
%%BeginResource: CMap (UniJ-UCS2-H)
%%Title: (UniJ-UCS2-H Adobe Japan1 2)
%%Version: 1

/CIDInit /ProcSet findresource begin

12 dict begin

begincmap

/CIDSystemInfo 3 dict dup begin
  /Registry (Adobe) def
  /Ordering (Japan1) def
  /Supplement 2 def
end def

/CMapName /UniJ-UCS2-H def
/CMapVersion 1 def
/CMapType 1 def

/WMode 0 def

1 begincodespacerange
  <0000> <FFFF>
endcodespacerange

100 begincidrange
<4e00> <4e00> 1200
<4e01> <4e01> 3000
<4e03> <4e03> 2275
<4e07> <4e07> 3754
<4e08> <4e08> 2510
<4e09> <4e09> 2174
<4e0a> <4e0a> 2509
<4e0b> <4e0b> 1340
<4e0d> <4e0d> 3526
<4e0e> <4e0e> 3881
<4e10> <4e10> 4091
<4e11> <4e11> 1233
<4e14> <4e14> 1484
<4e15> <4e15> 4092
<4e16> <4e16> 2632
<4e17> <4e17> 4311
<4e18> <4e18> 1648
<4e19> <4e19> 3594
<4e1e> <4e1e> 2511
<4e21> <4e21> 3974
<4e26> <4e26> 3602
<4e28> <4e28> 8371
<4e2a> <4e2a> 4093
<4e2d> <4e2d> 2980
<4e31> <4e31> 4094
<4e32> <4e32> 1778
<4e36> <4e36> 4095
<4e38> <4e38> 1561
<4e39> <4e39> 2926
<4e3b> <4e3b> 2323
<4e3c> <4e3c> 4096
<4e3f> <4e3f> 4097
<4e42> <4e42> 4098
<4e43> <4e43> 3307
<4e45> <4e45> 1649
<4e4b> <4e4b> 3309
<4e4d> <4e4d> 3259
<4e4e> <4e4e> 1911
<4e4f> <4e4f> 3681
<4e55> <4e55> 6480
<4e56> <4e56> 4099
<4e57> <4e57> 2512
<4e58> <4e58> 4100
<4e59> <4e59> 1333
<4e5d> <4e5d> 1757
<4e5e> <4e5e> 1956
<4e5f> <4e5f> 3829
<4e62> <4e62> 4659
<4e71> <4e71> 3930
<4e73> <4e73> 3285
<4e7e> <4e7e> 1505
<4e80> <4e80> 1615
<4e82> <4e82> 4101
<4e85> <4e85> 4102
<4e86> <4e86> 3971
<4e88> <4e88> 3879
<4e89> <4e89> 2794
<4e8a> <4e8a> 4104
<4e8b> <4e8b> 2244
<4e8c> <4e8c> 3275
<4e8e> <4e8e> 4107
<4e91> <4e91> 1248
<4e92> <4e92> 1939
<4e94> <4e94> 1938
<4e95> <4e95> 1194
<4e98> <4e98> 4081
<4e99> <4e99> 4080
<4e9b> <4e9b> 2083
<4e9c> <4e9c> 1125
<4e9e> <4ea0> 4108
<4ea1> <4ea1> 3682
<4ea2> <4ea2> 4111
<4ea4> <4ea4> 1958
<4ea5> <4ea5> 1195
<4ea6> <4ea6> 3744
<4ea8> <4ea8> 1686
<4eab> <4eac> 1687
<4ead> <4ead> 3070
<4eae> <4eae> 3972
<4eb0> <4eb0> 4112
<4eb3> <4eb3> 4113
<4eb6> <4eb6> 4114
<4eba> <4eba> 2579
<4ec0> <4ec0> 2372
<4ec1> <4ec1> 2580
<4ec2> <4ec2> 4119
<4ec4> <4ec4> 4117
<4ec6> <4ec6> 4118
<4ec7> <4ec7> 1650
<4eca> <4eca> 2067
<4ecb> <4ecb> 1392
<4ecd> <4ecd> 4116
<4ece> <4ece> 4115
<4ecf> <4ecf> 3577
<4ed4> <4ed4> 2196
<4ed5> <4ed5> 2195
<4ed6> <4ed6> 2846
<4ed7> <4ed7> 4120
<4ed8> <4ed8> 3527
<4ed9> <4ed9> 2699
endcidrange
```

< 6,431 CID ranges omitted >

```
endcmap
CMapName currentdict /CMap defineresource pop
end
end

%%EndResource
%%EOF
```



A CJK Solution

- Create Unicode ordering, including extra character forms
  - Example: Adobe-Unicode1-0
- Create CMaps specific for Japan, Korea, and PRC
  - Example: Japan-UCS2-H
  - Example: Korea-UCS2-H
  - Example: PRC-UCS2-H

A CJK solution based upon the Adobe CMap and CIDFont File Specification is relatively simple. This solution involves additional character forms, and requires that a totally new Registry-Ordering-Supplement be formed. For the purpose of this presentation, I call this character collection "Adobe-Unicode1-0," and include support only for those 94 kanji from JIS X 0208-1990.

This collection of characters contains not only the 94 kanji from JIS X 0208-1990, but also the 53 additional character forms necessary for Chinese and Korean markets. The following page provides a printout of the characters and their CIDs. Note how this collection is treated as a simple enumeration.

Next, I would need to craft a set of CMap files that map these characters to Unicode values. I created one for each Japanese, Chinese, and Korean.

The CMap entries for encoding-to-CID mappings are sparse only because I chose a set of characters that are spread throughout the Unicode kanji set. A complete Unicode kanji set would not be so sparse.

These three CMap files are found on the following pages.

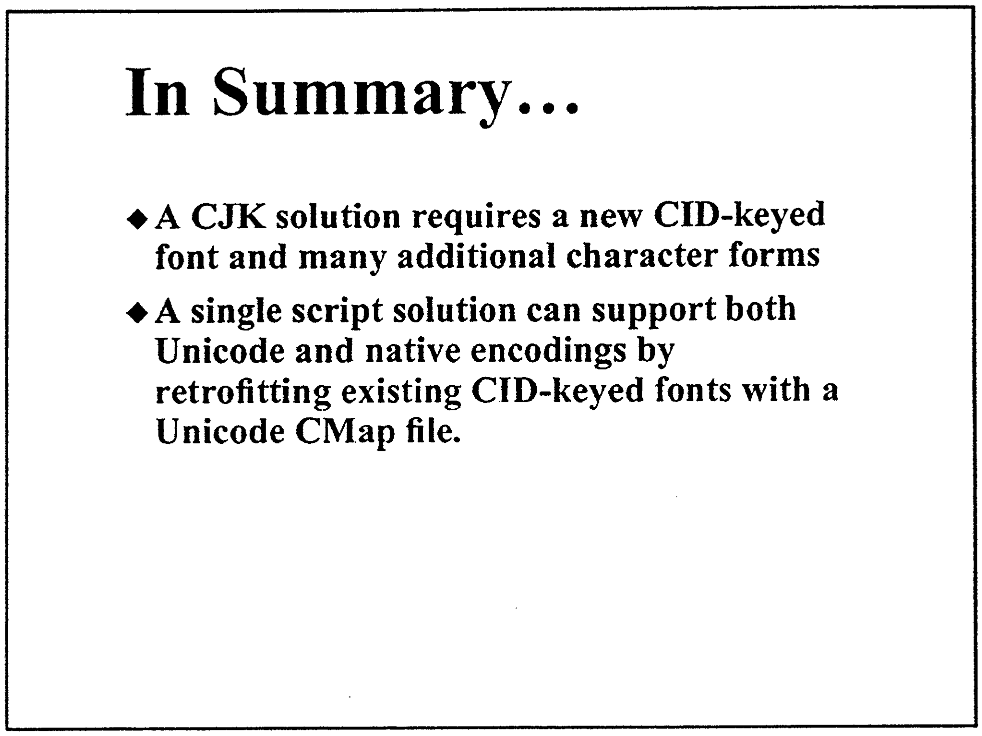
Adobe-Unicode1-0 CID-keyed Font

0	<SP>	31	尉	62	粟	93	鯵	124	畏
1	一	32	庵	63	絢	94	鰯	125	育
2	井	33	引	64	維	95	亥	126	胤
3	亜	34	悪	65	綾	96	以	127	芦
4	亥	35	惟	66	緯	97	位	128	茨
5	以	36	意	67	育	98	依	129	萎
6	伊	37	愛	68	胃	99	允	130	葵
7	位	38	慰	69	胤	100	医	131	虻
8	依	39	或	70	芋	101	印	132	衣
9	偉	40	扱	71	芦	102	哀	133	袷
10	允	41	按	72	茜	103	因	134	逸
11	医	42	挨	73	茨	104	夷	135	鞍
12	印	43	握	74	萎	105	姐	136	扱
13	咽	44	斡	75	葦	106	委	137	暗
14	哀	45	旭	76	葵	107	姻	138	淫
15	員	46	易	77	蔭	108	威	139	溢
16	唖	47	暗	78	虻	109	娃	140	磯
17	因	48	杏	79	衣	110	安	141	芦
18	囲	49	案	80	袷	111	宛	142	茨
19	圧	50	梓	81	謂	112	庵	143	逢
20	域	51	椅	82	逢	113	引	144	逸
21	壱	52	淫	83	逸	114	惟	145	違
22	夷	53	渥	84	違	115	意	146	遺
23	姐	54	溢	85	遺	116	按	147	飴
24	委	55	為	86	郁	117	挨		
25	姶	56	畏	87	闇	118	握		
26	姻	57	異	88	阿	119	暗		
27	威	58	磯	89	鞍	120	案		
28	娃	59	移	90	飲	121	梓		
29	安	60	稲	91	飴	122	渥		
30	宛	61	穐	92	鮎	123	溢		

```
%!PS-Adobe-3.0 Resource-CMap
%%DocumentNeededResources: procset CIDInit
%%IncludeResource: procset CIDInit
%%BeginResource: CMap (Japan-UCS2-H)
%%Title: (Japan-UCS2-H Adobe Unicode1 0)
%%Version: 1

/CIDInit /ProcSet findresource begin

12 dict begin

begincmap

/CIDSystemInfo 3 dict dup begin
  /Registry (Adobe) def
  /Ordering (Unicode1) def
  /Supplement 0 def
end def

/CMapName /Japan-UCS2-H def
/CMapVersion 1 def
/CMapType 1 def

/WMode 0 def

1 begincodespacerange
  <0000> <FFFF>
endcodespacerange

94 begincidrange
<4E00> <4E00>  1
<4E95> <4E95>  2
<4E9C> <4E9C>  3
<4EA5> <4EA5>  4
<4EE5> <4EE5>  5
<4F0A> <4F0A>  6
<4F4D> <4F4D>  7
<4F9D> <4F9D>  8
<5049> <5049>  9
<5141> <5141> 10
<533B> <533B> 11
<5370> <5370> 12
<54BD> <54BD> 13
<54C0> <54C0> 14
<54E1> <54E1> 15
<5516> <5516> 16
<56E0> <56E0> 17
<56F2> <56F2> 18
<5727> <5727> 19
<57DF> <57DF> 20
<58F1> <58F1> 21
<5937> <5937> 22
<59D0> <59D0> 23
<59D4> <59D4> 24
<59F6> <59F6> 25
<59FB> <59FB> 26
<5A01> <5A01> 27
<5A03> <5A03> 28
<5B89> <5B89> 29
<5B9B> <5B9B> 30
<5C09> <5C09> 31
<5EB5> <5EB5> 32
<5F15> <5F15> 33
<60AA> <60AA> 34
<60DF> <60DF> 35
<610F> <610F> 36
<611B> <611B> 37
<6170> <6170> 38
<6216> <6216> 39
<6271> <6271> 40
<6309> <6309> 41
<6328> <6328> 42
<63E1> <63E1> 43
<65A1> <65A1> 44
<65ED> <65ED> 45
<6613> <6613> 46
<6697> <6697> 47
<674F> <674F> 48
<6848> <6848> 49
<6893> <6893> 50
<6905> <6905> 51
<6DEB> <6DEB> 52
<6E25> <6E25> 53
<6EA2> <6EA2> 54
<70BA> <70BA> 55
<754F> <754F> 56
<7570> <7570> 57
<78EF> <78EF> 58
<79FB> <79FB> 59
<7A32> <7A32> 60
<7A50> <7A50> 61
<7C9F> <7C9F> 62
<7D62> <7D62> 63
<7DAD> <7DAD> 64
<7DBE> <7DBE> 65
<7DEF> <7DEF> 66
<80B2> <80B2> 67
<80C3> <80C3> 68
<80E4> <80E4> 69
<828B> <828B> 70
<82A6> <82A6> 71
<831C> <831C> 72
<8328> <8328> 73
<840E> <840E> 74
<8466> <8466> 75
<8475> <8475> 76
<852D> <852D> 77
<867B> <867B> 78
<8863> <8863> 79
<88B7> <88B7> 80
<8B02> <8B02> 81
<9022> <9022> 82
<9038> <9038> 83
<9055> <9055> 84
<907A> <907A> 85
<90C1> <90C1> 86
<95C7> <95C7> 87
<963F> <963F> 88
<978D> <978D> 89
<98F2> <98F2> 90
<98F4> <98F4> 91
<9B8E> <9B8E> 92
<9BF5> <9BF5> 93
<9C2F> <9C2F> 94
endcidrange
endcmap
CMapName currentdict /CMap defineresource pop
end
end

%%EndResource
%%EOF
```

```
%!PS-Adobe-3.0 Resource-CMap
%%DocumentNeededResources: procset CIDInit
%%IncludeResource: procset CIDInit
%%BeginResource: CMap (PRC-UCS2-H)
%%Title: (PRC-UCS2-H Adobe Unicode1 0)
%%Version: 1

/CIDInit /ProcSet findresource begin

12 dict begin

begincmap

/CIDSystemInfo 3 dict dup begin
  /Registry (Adobe) def
  /Ordering (Unicode1) def
  /Supplement 0 def
end def

/CMapName /PRC-UCS2-H def
/CMapVersion 1 def
/CMapType 1 def

/WMode 0 def

1 begincodespacerange
  <0000> <FFFF>
endcodespacerange

94 begincidrange
<4E00> <4E00>   1
<4E95> <4E95>   2
<4E9C> <4E9C>   3
<4EA5> <4EA5>  95 % replacement
<4EE5> <4EE5>  96 % replacement
<4F0A> <4F0A>   6
<4F4D> <4F4D>  97 % replacement
<4F9D> <4F9D>  98 % replacement
<5049> <5049>   9
<5141> <5141>  99 % replacement
<533B> <533B> 100 % replacement
<5370> <5370> 101 % replacement
<54BD> <54BD>  13
<54C0> <54C0> 102 % replacement
<54E1> <54E1>  15
<5516> <5516>  16
<56E0> <56E0> 103 % replacement
<56F2> <56F2>  18
<5727> <5727>  19
<57DF> <57DF>  20
<58F1> <58F1>  21
<5937> <5937> 104 % replacement
<59D0> <59D0> 105 % replacement
<59D4> <59D4> 106 % replacement
<59F6> <59F6>  25
<59FB> <59FB> 107 % replacement
<5A01> <5A01> 108 % replacement
<5A03> <5A03> 109 % replacement
<5B89> <5B89> 110 % replacement
<5B9B> <5B9B> 111 % replacement
<5C09> <5C09>  31
<5EB5> <5EB5> 112 % replacement
<5F15> <5F15> 113 % replacement
<60AA> <60AA>  34
<60DF> <60DF> 114 % replacement
<610F> <610F> 115 % replacement
<611B> <611B>  37
<6170> <6170>  38
<6216> <6216>  39
<6271> <6271>  40
<6309> <6309> 116 % replacement
<6328> <6328> 117 % replacement
<63E1> <63E1> 118 % replacement
<65A1> <65A1>  44
<65ED> <65ED>  45
<6613> <6613>  46
<6697> <6697> 119 % replacement
<674F> <674F>  48
<6848> <6848> 120 % replacement
<6893> <6893> 121 % replacement
<6905> <6905>  51
<6DEB> <6DEB>  52
<6E25> <6E25> 122 % replacement
<6EA2> <6EA2> 123 % replacement
<70BA> <70BA>  55
<754F> <754F> 124 % replacement
<7570> <7570>  57
<78EF> <78EF>  58
<79FB> <79FB>  59
<7A32> <7A32>  60
<7A50> <7A50>  61
<7C9F> <7C9F>  62
<7D62> <7D62>  63
<7DAD> <7DAD>  64
<7DBE> <7DBE>  65
<7DEF> <7DEF>  66
<80B2> <80B2> 125 % replacement
<80C3> <80C3>  68
<80E4> <80E4> 126 % replacement
<828B> <828B>  70
<82A6> <82A6> 127 % replacement
<831C> <831C>  72
<8328> <8328> 128 % replacement
<840E> <840E> 129 % replacement
<8466> <8466>  75
<8475> <8475> 130 % replacement
<852D> <852D>  77
<867B> <867B> 131 % replacement
<8863> <8863> 132 % replacement
<88B7> <88B7> 133 % replacement
<8B02> <8B02>  81
<9022> <9022>  82
<9038> <9038> 134 % replacement
<9055> <9055>  84
<907A> <907A>  85
<90C1> <90C1>  86
<95C7> <95C7>  87
<963F> <963F>  88
<978D> <978D> 135 % replacement
<98F2> <98F2>  90
<98F4> <98F4>  91
<9B8E> <9B8E>  92
<9BF5> <9BF5>  93
<9C2F> <9C2F>  94
endcidrange
endcmap
CMapName currentdict /CMap defineresource pop
end
end

%%EndResource
%%EOF
```

```
%!PS-Adobe-3.0 Resource-CMap
%%DocumentNeededResources: procset CIDInit
%%IncludeResource: procset CIDInit
%%BeginResource: CMap (Korea-UCS2-H)
%%Title: (Korea-UCS2-H Adobe Unicode1 0)
%%Version: 1

/CIDInit /ProcSet findresource begin

12 dict begin

begincmap

/CIDSystemInfo 3 dict dup begin
  /Registry (Adobe) def
  /Ordering (Unicode1) def
  /Supplement 0 def
end def

/CMapName /Korea-UCS2-H def
/CMapVersion 1 def
/CMapType 1 def

/WMode 0 def

1 begincodespacerange
  <0000> <FFFF>
endcodespacerange

94 begincidrange
<4E00> <4E00>   1
<4E95> <4E95>   2
<4E9C> <4E9C>   3
<4EA5> <4EA5>   4
<4EE5> <4EE5>   5
<4F0A> <4F0A>   6
<4F4D> <4F4D>   7
<4F9D> <4F9D>   8
<5049> <5049>   9
<5141> <5141>  10
<533B> <533B>  11
<5370> <5370>  12
<54BD> <54BD>  13
<54C0> <54C0>  14
<54E1> <54E1>  15
<5516> <5516>  16
<56E0> <56E0>  17
<56F2> <56F2>  18
<5727> <5727>  19
<57DF> <57DF>  20
<58F1> <58F1>  21
<5937> <5937>  22
<59D0> <59D0>  23
<59D4> <59D4>  24
<59F6> <59F6>  25
<59FB> <59FB>  26
<5A01> <5A01>  27
<5A03> <5A03>  28
<5B89> <5B89>  29
<5B9B> <5B9B>  30
<5C09> <5C09>  31
<5EB5> <5EB5>  32
<5F15> <5F15>  33
<60AA> <60AA>  34
<60DF> <60DF>  35
<610F> <610F>  36
<611B> <611B>  37
<6170> <6170>  38
<6216> <6216>  39
<6271> <6271> 136 % replacement
<6309> <6309>  41
<6328> <6328>  42
<63E1> <63E1>  43
<65A1> <65A1>  44
<65ED> <65ED>  45
<6613> <6613>  46
<6697> <6697> 137 % replacement
<674F> <674F>  48
<6848> <6848>  49
<6893> <6893>  50
<6905> <6905>  51
<6DEB> <6DEB> 138 % replacement
<6E25> <6E25>  53
<6EA2> <6EA2> 139 % replacement
<70BA> <70BA>  55
<754F> <754F>  56
<7570> <7570>  57
<78EF> <78EF> 140 % replacement
<79FB> <79FB>  59
<7A32> <7A32>  60
<7A50> <7A50>  61
<7C9F> <7C9F>  62
<7D62> <7D62>  63
<7DAD> <7DAD>  64
<7DBE> <7DBE>  65
<7DEF> <7DEF>  66
<80B2> <80B2>  67
<80C3> <80C3>  68
<80E4> <80E4>  69
<828B> <828B>  70
<82A6> <82A6> 141 % replacement
<831C> <831C>  72
<8328> <8328> 142 % replacement
<840E> <840E>  74
<8466> <8466>  75
<8475> <8475>  76
<852D> <852D>  77
<867B> <867B>  78
<8863> <8863>  79
<88B7> <88B7>  80
<8B02> <8B02>  81
<9022> <9022> 143 % replacement
<9038> <9038> 144 % replacement
<9055> <9055> 145 % replacement
<907A> <907A> 146 % replacement
<90C1> <90C1>  86
<95C7> <95C7>  87
<963F> <963F>  88
<978D> <978D>  89
<98F2> <98F2>  90
<98F4> <98F4> 147 % replacement
<9B8E> <9B8E>  92
<9BF5> <9BF5>  93
<9C2F> <9C2F>  94
endcidrange
endcmap
CMapName currentdict /CMap defineresource pop
end
end

%%EndResource
%%EOF
```



In Summary...

- ◆ A CJK solution requires a new CID-keyed font and many additional character forms
- ◆ A single script solution can support both Unicode and native encodings by retrofitting existing CID-keyed fonts with a Unicode CMap file.

The bottom line is that a CJK solution, namely a single font that satisfies Japanese, Chinese, and Korean users, require additional character forms. Substituting these character forms becomes a trivial task for CID technology.

Second, adding Unicode support to existing CID-keyed fonts, such as Japanese, becomes a simple matter of retrofitting through the issuance of a new CMap—the CID-keyed font remains unchanged.

For more detailed information on CID technology, I suggest contacting Adobe Systems' Developer Support to request one or more of the following documents:

Technical Specification #5014: "The CMap and CID-keyed Font Files Specification"

Technical Specification #5078: "Adobe-Japan1-2 Character Collection for CID-keyed Fonts"

Technical Specification #5079: "Adobe-GB1-0 Character Collection for CID-keyed Fonts"

Technical Specification #5080: "Adobe-BigFive1-0 Character Collection for CID-keyed Fonts"

Technical Note #5092: "CID-keyed Font File Format Overview"

Technical Specification #5093: "Adobe-Korea1-0 Character Collection for CID-keyed Fonts"