1. 파일 시그니처란? + Header와 Footer 알아보기 [PDF, PNG, JPF, JPEG, GIF, ZIP, MP3, BMP]

파일 시그니처 : 파일 형식을 식별하기 위해 파일 내부에 저장된 특정한 byte 단위의 시그니처가 파일의 처음이나 끝에 존재하는 포맷 / 파일의 처음에 있으면 헤더(Header), 파일 끝에 있으면 푸터(Footer)

파일 형식	Header	Footer
PDF	25 50 44 46 (ASCII: %PDF)	25 25 45 4F 46 (ASCII: %%EOF)
PNG	89 50 4E, 47 0D 0A 1A 0A	49 45 4E 44 AE 42 60 82
JPG/JPEG	FF D8 FF E0, FF D8 FF E1, FF D8 FF E8	FF D9
GIF	47 49 46 38 37 61(GIF87a), 47 49 46 38 39 61 (GIF89a)	00 3B
ZIP	50 4B 03 04	50 4B 05 06
MP3	FF FB, FF F3, FF F2	
BMP	42 4D (ASCII: BM)	

2. 문자 코드 [아스키 코드/엡시딕 코드/유니코드] + Base64란?

아스키 코드 : 7비트로 영문 대소문자, 숫자, 특수문자를 포함하여 128개 문자를 표현 가능하며 8비트 확장 버전은 256개 문자를 지원. 제어 문자와 그래픽 문자로 구분되며 통신 및 출력 제어에 활용.

엡시딕 코드 : 8비트 문자 인코딩 방식으로 IBM 시스템에서 주로 사용하며 아스키 코드와 호환성 낮음

유니코드 : 2바이트 이상으로 전 세계 모든 문자를 일관되게 표현하는 국제 문자 인코딩의 표준. UTF-8, UTF-16, UTF-32 등 다양한 인코딩 방법을 제공

3. 빅엔디안과 리틀엔디안

엔디안 : 데이터에 여러 바이트가 있는 경우 바이트의 순서를 정하는 규칙

- 빅엔디안
  낮은 주소에 데이터의 높은 바이트부터 저장되어 숫자의 상위 바이트가 최하위 주소에 저장되고 하위 바이트가 최상위 주소에 저장. 평소 사람이 사용하는 선형 방식과 같아서 메모리 저장된 순서 그대로 읽을 수 있으며 이해하기 쉬움
- 리틀엔디안
  낮은 주소에 데이터의 낮은 바이트를 저장되어 숫자의 하위 바이트가 메모리의 최하위 주소에 저장되고 상위 바이트가 최상위 주소에 저장. 평소 사람이 숫자를 사용하는 선형 방식과 반대로 읽어야 함.

X86 아키텍처가 리틀 엔디안 쓰기에 오늘날 대부분의 데스크톱 컴퓨터는 리틀 엔디안 방식을 사용. 전통적인 os는 주로 빅 엔디안 사용. 네트워크 전송 시 엔디안이 다를 경우에는 데이터 해설 순서가 달라져서 값이 이상하게 바뀔 가능성 존재.